

INFOWATCH

InfoWatch Traffic Monitor 7.13

Administration Guide

08/03/2026

© JSC “InfoWatch”

Tel/Fax +7 (495) 229-00-22

http://www.infowatch.com

TABLE OF CONTENTS

1 Introduction 8
1.1 Who Should Read this Guide 8
1.2 Set of Documents 8
1.3 Technical Support 8
2 Traffic Monitor Overview 9
2.1 InfoWatch Traffic Monitor Features 9
2.1.1 In Stream/At The Gateway Traffic Interception 9
SMTP Traffic Interception Scheme 9
POP3 Traffic Interception Scheme 12
HTTP Traffic Interception Scheme 13
HTTPS Traffic Interception Scheme 16
ICQ Traffic Interception Scheme 18
NRPC Traffic Interception Scheme 19
IMAP4 Traffic Interception Scheme 20
2.1.2 Processing Traffic, Received from Third-party Systems 20
2.2 InfoWatch Traffic Monitor Subsystems (Red Hat Enterprise Linux) 21
2.3 Licensing 22
3 System Installation Types 27
4 Configuring System After Installation 28
4.1 Changing the Preset Password 28
4.2 Configuring protection against password guessing 30
4.3 Preliminary Settings 31
4.3.1 Configuring Time Synchronization 31
4.3.2 Configuring Sphinx for Distributed Installation 32
4.4 General Settings 32
4.4.1 Configuring Server to Work with SPAN, or Port Mirroring Technology 32
Configuring Traffic Monitor Server as Sniffer 33
Configuring Traffic Monitor Server to Receive Copy of Traffic from Sniffer 34
Creating Traffic Monitor Cluster 35
4.4.2 Intercepting Traffic, Transmitted via ICAP Protocol 36
Configuring ICAP 37
Recommendations on SQUID Configuration 38
Recommendations on configuring UserGate 43

Disabling ICAP ... 46
4.4.3 Configuring Licensing Server (in case of Cluster) ... 46
4.5 Configuring Traffic Interception ... 47
4.5.1 Configuring SMTP Traffic Interception ... 48
SPAN, or Port Mirroring ... 48
Receiving Copies from Mail Server ... 49
Inline Mode ... 49
Configuring Mail Servers for SMTP Traffic Interception ... 52
4.5.2 Configuring POP3 Traffic Interception ... 58
4.5.3 Configuring HTTP Traffic Interception ... 58
SPAN, or Port Mirroring ... 58
Inline Mode ... 58
4.5.4 Configuring HTTPS Traffic Interception ... 59
Inline Mode ... 59
4.5.5 Configuring ICQ Traffic Interception ... 59
SPAN, or Port Mirroring ... 59
4.5.6 Configuring NRPC Traffic Interception ... 60
SPAN, or Port Mirroring ... 60
4.5.7 Configuring IMAP4 Traffic Interception ... 60
4.5.8 Receiving Objects Intercepted by InfoWatch Device Monitor ... 61
4.6 Process autorun ... 61
4.6.1 Check of Process Autorun ... 61
4.6.2 Enabling and Disabling Process Autorun ... 64
4.7 Remote database interaction module ... 66
4.7.1 Configuring Data Collection in a Branch Network ... 66
4.7.2 Configuring the Client of the Remote Database Management Module ... 66
4.7.3 Configuring the Server of the Remote Database Management Module ... 67
4.8 Configuring OCR Extractors ... 68
4.8.1 Interception channels ... 68
4.8.2 Supported OCR extractors ... 70
Tesseract settings ... 70
ABBYY FineReader Engine settings ... 71
4.9 Configuring Notifications for Users and Employees ... 73
4.10 Limiting the number of events that can be found ... 74
4.11 Service-to-service cooperation setup (Consul service) ... 74
4.11.1 Starting and stopping the service ... 74
4.11.2 Registering services in Consul ... 75
4.11.3 Distributed installation ... 76
4.11.4 Network rules configuration to access Consul ... 78
4.11.5 consul.json configuration file and iwtm-consul.service unit file ... 78
4.12 Secure Deployment Recommendations ... 81

4.12.1 Components ports 81
4.13 Configuring Verdict Management 87
5 Settings for automatic deletion of Data Discovery events 88
6 Monitoring 89
6.1 Monitoring Subsystem for Distributed Architecture 89
6.1.1 Configuring Connection to Device Monitor 89
6.1.2 Configuring Indicators Manually 90
6.1.3 Configuring Addresses of Time Synchronization Server for Monitoring Subsystem 90
6.1.4 Configuring Notifications 90
7 Using a database unloading for automatic recipient validataion 92
8 Database Administration 94
8.1 Oracle 94
8.1.1 Tablespaces in InfoWatch Traffic Monitor database 94
General information 94
Storing 96
Archiving 97
Deleting 97
8.1.2 Managing Daily Tablespaces 97
Configuring Location of Files within File System 97
Configuring Storage Modes for Tablespace Files 99
Archiving Daily Tablespaces 101
Restoring Daily Tablespaces 105
Deleting Daily Tablespaces 106
Managing Daily Tablespaces with the Console Utility 109
8.1.3 Load statistics of Daily Tablespaces 111
Scenarios of script execution 112
8.1.4 Database Backup 115
Creating Database Backup 116
Restoring the Database from the Backup 124
8.1.5 Database Routinely Maintenance 127
8.1.6 Database Logging 129
Rotation of database logs 129
8.2 PostgreSQL 130
8.2.1 Tablespaces in InfoWatch Traffic Monitor database 130
General information 130
Storing 132
Archiving 133
Deleting 133

8.2.2 Managing Daily Tablespaces ... 133
Configuring Location of Files in File System ... 133
Configuring Storage Modes for Tablespace Files ... 135
Archiving Daily Tablespaces ... 138
Restoring Daily Tablespaces ... 142
Deleting Daily Tablespaces ... 143
Managing Daily Tablespaces with the Console Utility ... 146
8.2.3 Load statistics of Daily Tablespaces ... 147
Scenarios of script execution ... 149
8.2.4 Database Backup ... 152
Creating Database Backup ... 153
Restoring Database from Backup ... 159
8.2.5 Database Server Routinely Maintenance ... 162
8.2.6 Database logging ... 163
Rotation of database logs ... 164

9 Administrating the Server End of InfoWatch Traffic Monitor ... 165

9.1 Traffic Monitor Server Processes ... 165
9.1.1 List of Traffic Monitor Server End Processes ... 165
9.1.2 Setting Traffic Monitor Server End Configuration Files ... 170
9.1.3 Using Traffic Monitor Server End Processes ... 171

9.2 Configuring OCR ... 173
9.2.1 Enabling OCR technology when implementing the System ... 174
9.2.2 Configuring OCR on several nodes ... 174
9.2.3 Configuration File ocr_custom.xml ... 174

9.3 Configuring Settings for Processing Enclosed Archives ... 176
9.3.1 The extractors.xml Configuration File ... 176
9.3.2 The extractors_customization.xml Configuration File ... 180

9.4 Configuring Settings of Search Using Wildcards ... 182

9.5 Configuring Integration with Data Analysis Service ... 183

9.6 Configuring Connection with Data Storage ... 186

9.7 Logging System Work ... 188

9.8 File Queues ... 189

9.9 Recovering the System after Failures ... 192

9.10 Managing Languages Including Morphological Support ... 193
9.10.1 Adding New Language for Events Search. Morphology and Adding Terms. ... 193
9.10.2 Updating Installed Language Including Morphological Support ... 194
9.10.3 Deleting Search Language and Terms ... 195

9.11 Configuring Forwarding Data to SIEM ... 195
9.11.1 Configuring SIEM Side ... 196

Table Representation of TM Events ... 196
User Audit Table View ... 202
9.11.2 Configuring TM Side ... 207
Managing Siem User ... 207
Forwarding Logs to SIEM ... 209
Log Management for TM DB User Sessions ... 209
9.11.3 Type of Logs, Transmitted to SIEM ... 210
9.12 Deleting Temporary Files ... 210
10 Configuring Single Sign-On in InfoWatch products ... 212
11 Interaction with external systems ... 214
11.1 Uploading events (PushAPI SDK) ... 214
11.1.1 Principles of using PushAPI SDK programming interface ... 214
11.1.2 General description of PushAPI SDK programming interface ... 217
Data Types in PushAPI SDK ... 218
Exceptions of PushAPI SDK ... 220
Traffic Monitor PushAPI SDK services ... 221
Contact types in Traffic Monitor PushAPI SDK ... 223
Traffic Monitor PushAPI SDK supported attributes ... 223
EventProcessor service in PushAPI SDK ... 227
11.1.3 Registering external component with PushAPI SDK ... 229
PushAPI SDK third-party component registry file format ... 230
Updating the third-party component registry file ... 244
11.1.4 Description of PushAPI SDK samples ... 248
Console utilities ... 248
Building PushAPI SDK samples ... 250
Examples on How to Use the pushapi-util Utility ... 251
Building samples under Linux in PushAPI SDK ... 258
Building samples under Windows in PushAPI SDK ... 258
11.1.5 Troubleshooting When Sending Events ... 258
11.2 Uploading Events via HTTP Protocol ... 259
11.2.1 Uploading Events with Web Message Type ... 261
11.2.2 Uploading Events with Messenger Type ... 263
Uploading Calls ... 263
Uploading Text Messages ... 265
Uploading Files ... 267
11.2.3 Uploading Events with File Exchange Type ... 269
Copying File to Network Resource ... 269
Copying File to Removable Device ... 273
Copying File via Terminal Session ... 276
Sending File to Cloud Storage ... 280

Sending File via FTP 281
11.2.4 Uploading Events with Print Type 285
11.2.5 Uploading Events to Get Traffic Monitor Verdict 286
11.3 Uploading sample documents and DB unloadings (REST API SDK) 289
11.3.1 Configuring REST API programming interface 289
11.3.2 Registering REST API external component 290
REST API third-party component registry file format 290
11.3.3 Format of sample unloading and REST API sample document 293
11.3.4 Functional description of REST API programming interface 293
Common headings for GET/POST requests 294
Common errors 294
Working with configuration 302
Working with sample unloadings 302
Working with REST API sample documents 308
11.4 Access to InfoWatch Traffic Monitor events for external systems (DataExport API SDK) .. 322
11.4.1 DataExport API functionality 322
11.4.2 Authorization to access DataExport API 324
Work with plugins 324
Work with tokens 324
11.4.3 DataExport API third-party component registration 326
DataExport API third-party component registry file format 326
11.4.4 Functional description of DataExport API application programming interface 330
Common Headers of GET/POST Queries for DataExport API 331
Common errors 333
Getting list of features for the current version of Traffic Monitor 340
Retrieving audit data 340
Getting organizational structure 349
Getting events 390
Retrieving the list of Console users 428
Getting Results of Completed Traffic Monitor Query 429
12 Appendix A. Username and Password Creation Recommendations 433
13 Appendix B. Monitoring Indicators 435

1 Introduction

This guide provides information on InfoWatch Traffic Monitor administration: configuration of the system after installation, disaster recovery, routine maintenance.

1.1 Who Should Read this Guide

The guide is intended for InfoWatch Traffic Monitor Server administrators, which are familiar with the basics of Microsoft Windows and Linux and have the skills of Oracle and Postgre SQL DBMS administration.

1.2 Set of Documents

For more information see the following documents:

- «*InfoWatchTraffic Monitor. Installation Guide*».

The document describes how to install, configure, update, and uninstall the InfoWatch Traffic Monitor.

- «*InfoWatch Traffic Monitor. Administration Guide*».

Contains information on System administration (database, server-end).

- «*InfoWatch Traffic Monitor. User Guide*».

The document describes operation with InfoWatch Traffic Monitor: configuration setting, data export/import, creating policies for objects processing.

- «*InfoWatch Traffic Monitor. Configuration files guide*».

The document describes the frequently used configuration files.

In addition to the primary System User Guide and Administration Guide, a Knowledge Base is available at lk.infowatch.ru, which includes troubleshooting materials and answers to questions. If you do not have access to the Knowledge Base, please contact the technical support service of InfoWatch at support@infowatch.com.

1.3 Technical Support

If you cannot resolve issues which arise when using the System or have questions regarding the use of the System, you can contact the Technical Support Service:

- If you purchased the product from an InfoWatch partner, contact the Technical Support Service of this partner company.
- If you purchased the product directly from InfoWatch, you can contact our Technical Support Service at support@infowatch.com.

You can also visit online support at our website at: http://infowatch.com/support.

2 Traffic Monitor Overview

In this chapter:

- InfoWatch Traffic Monitor Features;
- InfoWatch Traffic Monitor Subsystems (Red Hat Enterprise Linux);
- Licensing.

2.1 InfoWatch Traffic Monitor Features

InfoWatch Traffic Monitor allows you to control corporate network information flows in order to detect and prevent unauthorized use of confidential data.

Main features of InfoWatch Traffic Monitor:

- In Stream/At The Gateway Traffic Interception – interception of traffic transferred in stream/at the gateway via SMTP, POP3, HTTP, HTTPS, OSCAR (ICQ), NRPC (IBM Notes);
- Traffic intercepted by Device Monitor agents on workstations (see in *InfoWatch Device Monitor. User Guide*);
- Processing Traffic, Received from Third-party Systems;
- Analyzing content of intercepted traffic in order to detect violation of the corporate security policy;
- Filtering intercepted traffic by allowing/prohibiting delivery of particular data.

2.1.1 In Stream/At The Gateway Traffic Interception

The chapter contains information about traffic types and their interception schemes:

- SMTP Traffic Interception Scheme;
- POP3 Traffic Interception Scheme;
- HTTP Traffic Interception Scheme;
- HTTPS Traffic Interception Scheme;
- ICQ Traffic Interception Scheme;
- NRPC Traffic Interception Scheme;
- IMAP4 Traffic Interception Scheme.

SMTP Traffic Interception Scheme

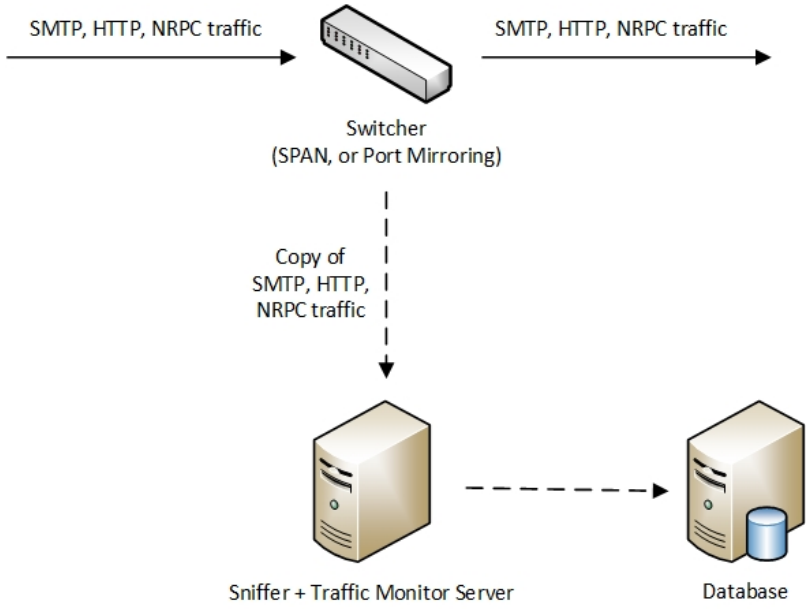
Method 1:

Traffic is being intercepted from a managed switch by using SPAN Port, or Port Mirroring technology.

Big number of managed network switches allows duplicating traffic from one or several ports or VLAN on a separate port.

In this case the customized switch port is being attached to Traffic Monitor Server network interface. Server interface is being switched to promiscuous mode, which allows accepting all incoming packages. This interface does not require an IP-address.

The most effective is to mirror internal switch port or internal proxy-server interface.



For more information on functionality *see SPAN, or Port Mirroring*.

Advantages:

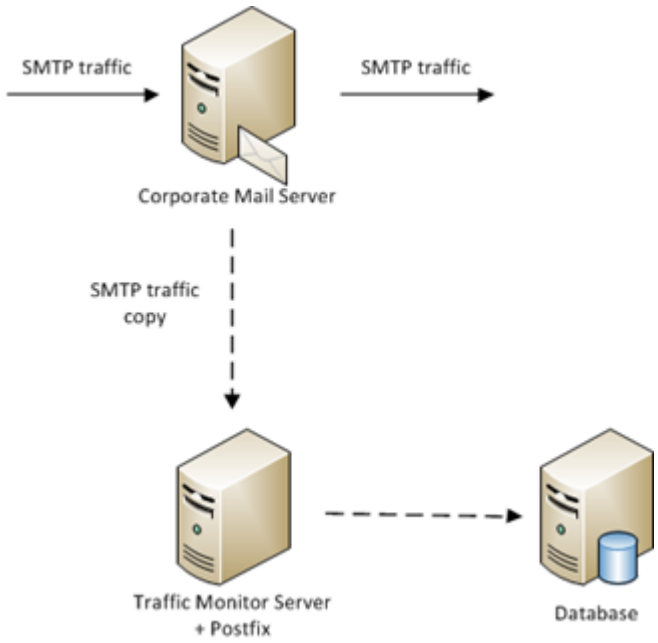
- This method allows to analyze traffic safely even in the isolated network segment. System can never influence analyzed traffic, regardless of Server state.
- SMTP traffic of external servers is being analyzed as well

Shortcomings:

- Under heavy load of switch, packages can be partially lost, especially, when the traffic is being mirrored from two ports to one, therefore we need:
  - switch with corresponding functionality;
  - additional network adapter on Traffic Monitor server.
- Among shortcomings, internal communications are not being controlled.

Method 2:

On a corporate mail server it is necessary to setup a rule, sending shadow copy (BCC) for each sent message. Copy should be sent to mail address with mail domain, which has the same IP-address as Traffic Monitor Server. Most of mail servers have such functionality.



For more information on functionality see *Receiving Copies from Mail Server*.

Advantages:

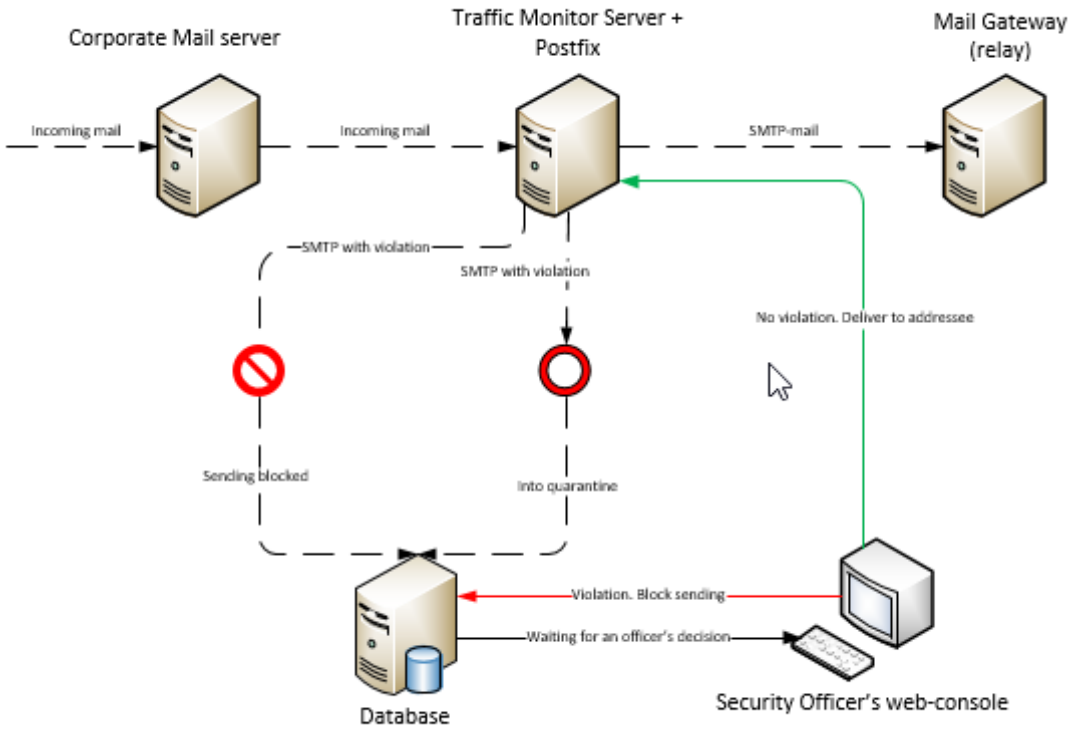
- Apart from external, it allows analyzing company's internal communications;
- When MS Exchange 2007 or newer is used there is functionality to analyze communications of a certain user group;
- Guarantees entire mail analysis.

Shortcomings:

- Requires settings of a corporate mail server to be changed;
- External mail servers (if allowed to use) is not being controlled;
- Upon Traffic Monitor Server inaccessibility, users will receive mail about shadow copy delivery error;
- Additional mail server load.

Method 3:

Traffic Monitor server uses configured Postfix mail server and functions as an SMTP retranslator.



For more information on the functionality see *Inline Mode*.

Advantages:

Allows enabling mail traffic block functionality.

Shortcomings:

- Requires settings of a corporate mail server to be changed;
- External mail servers (if allowed to use) is not being controlled;
- Upon Traffic Monitor server inaccessibility, users cannot send mail, in case server reservation has not been provided (for example, through MX-records).

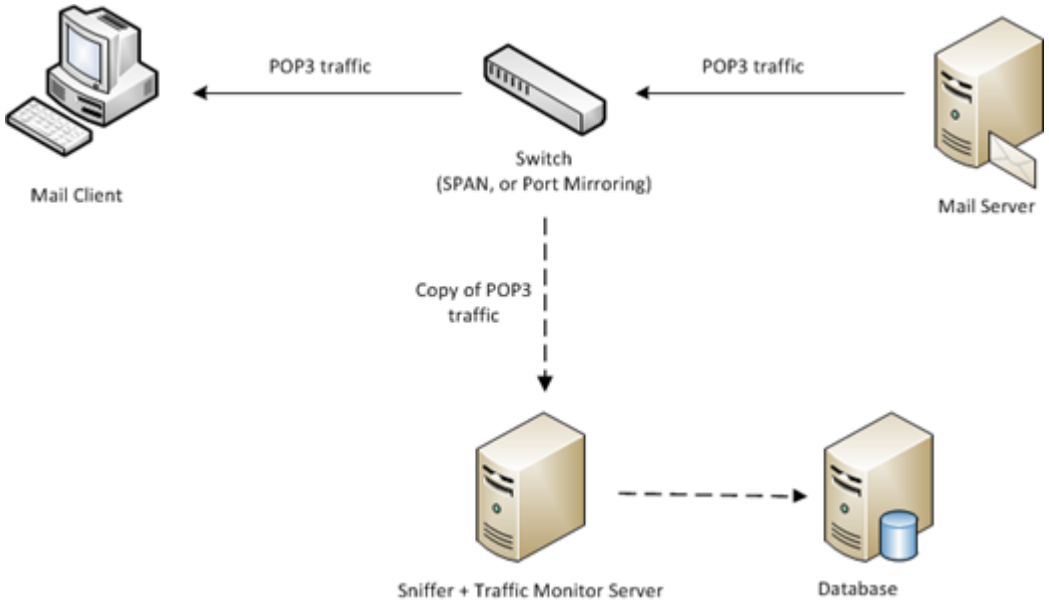
POP3 Traffic Interception Scheme

Traffic is intercepted from a managed switch by using SPAN Port, or Port Mirroring technology.

A large number of managed network switches allows duplicating traffic from one or several ports or VLAN to a single port.

In this case the customized switch port is connected to Traffic Monitor Server network interface. Server interface is switched to promiscuous mode, which allows accepting all incoming packages. This interface does not require an IP-address.

The most effective is to mirror the internal switch port or internal proxy-server interface.



For more information on functionality see *Configuring POP3 Traffic Interception*.

HTTP Traffic Interception Scheme

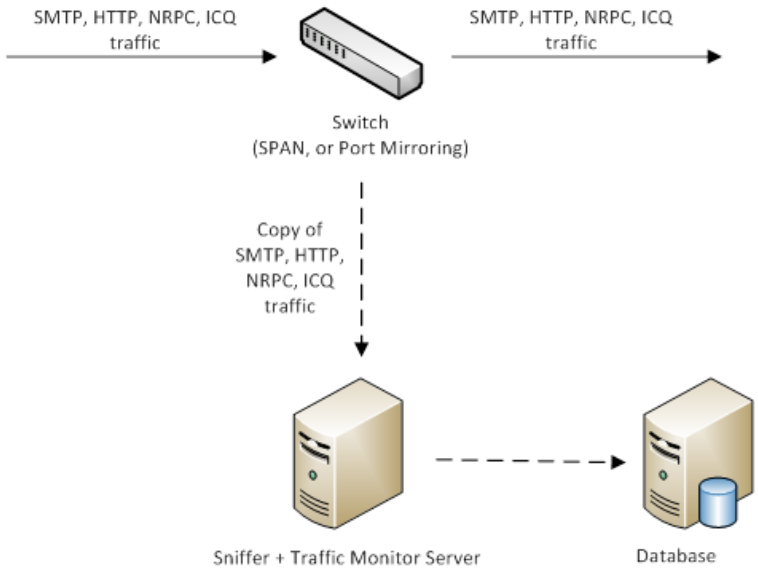
Method 1:

Traffic is intercepted from a managed switch by using SPAN Port, or Port Mirroring technology.

A large number of managed network switches allows duplicating traffic from one or several ports or VLAN to a single port.

In this case the customized switch port is connected to Traffic Monitor Server network interface. Server interface is switched to promiscuous mode, which allows accepting all incoming packages. This interface does not require an IP address.

The most effective is to mirror the internal switch port or internal proxy server interface.



For more information on functionality see *SPAN, or Port Mirroring*.

Advantages:

- This method allows to analyze traffic safely even in the isolated network segment. System can never influence the analyzed traffic, regardless of Server state.

Shortcomings:

- Under heavy load on the switch, some packages can be lost, especially, when the traffic is mirrored from two ports to a single port. To prevent the loss of packages, you should use:
  - a switch with corresponding functionality;
  - an additional network adapter on Traffic Monitor server.

Special case 1:

In case Traffic Monitor server is located in virtual environment, in addition to switch port setup, to analyze traffic an individual virtual switch is being created with promiscuous mode enabled. Switch should be connected to an individual physical interface. This method allows you to analyze the traffic of virtual machines running through a virtual switch.

This method works on VMware ESXi servers. On Hyper-V this functionality is implemented with limitations starting with version 2012 Hyper-V.

Special case 2:

Instead of managed switch, HUB can be used. This variant's disadvantage is HUB device's is low transmission capacity, as well as inability to buy a device in connection to removal from production.

Method 2:

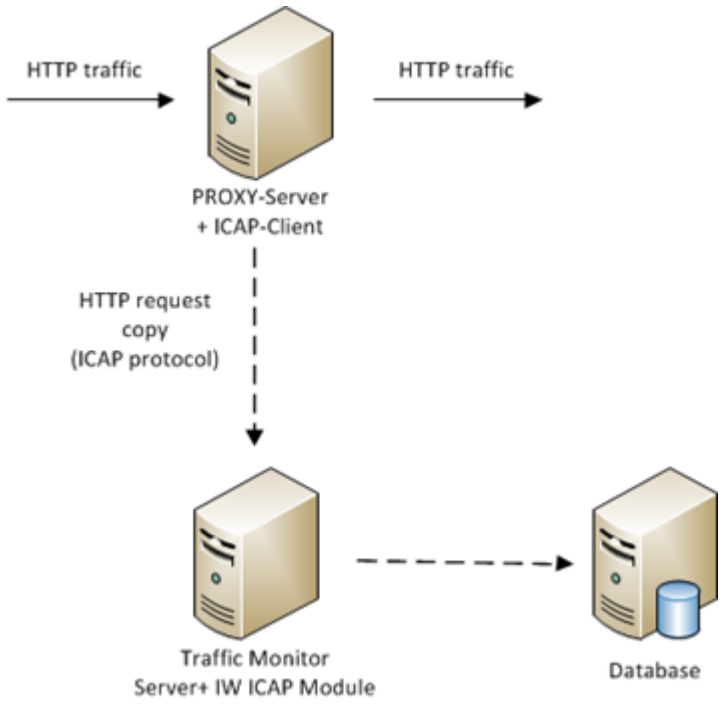
Traffic copy, which is transmitted via ICAP protocol, is intercepted from corporative proxy server.

Traffic Monitor Server is positioned as an ICAP server. In case there is a proxy server in company, which supports ICAP protocol, it is possible to use this functionality for analysis of HTTP traffic.

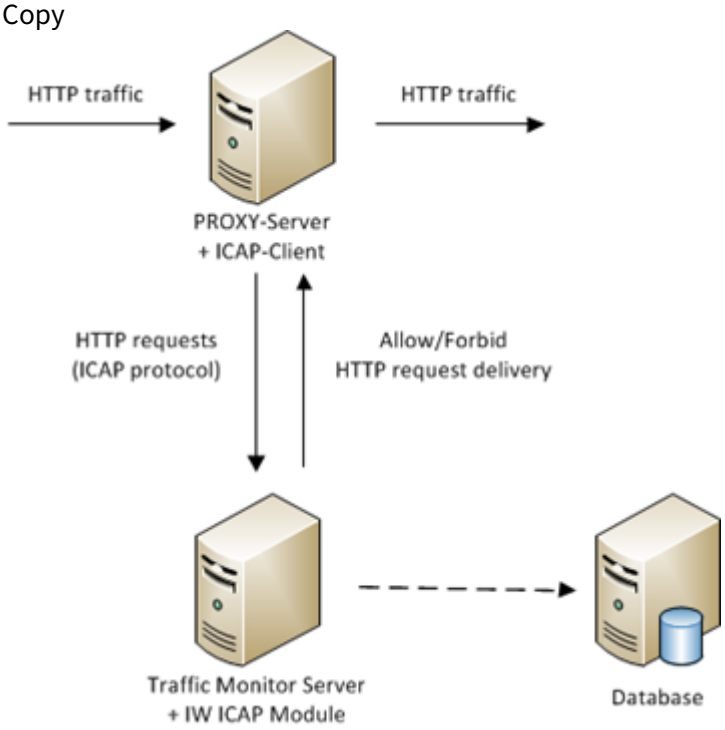
The following proxy servers support this option:

- SQUID (given that the package has been compiled including ICAP support)
- Zimbra Proxy (as a part of Zimbra ZCS)

> **Note:**
>
> To work with other proxy servers maintaining ICAP is possible, but this requires preliminary compatibility test.



Copy



Block

For more information on functionality see *Inline mode*.

Advantages:

- Guarantees all packages delivery;
- Traffic Monitor Server can work in blocking mode («inline») with some proxy servers.

Shortcomings:

- Proxy server with corresponding functionality is required;
- Additional proxy server load;

- Changing corporate proxy server setup;
- Slight increase in the response time of Internet resources is possible.

HTTPS Traffic Interception Scheme

> **Warning!**
>
> All methods of analyzing HTTPS traffic should be used with caution. The technology of analyzing HTTPS traffic is based on replacing a certificate of HTTPS session with a certificate issued by the proxy server (**man-in-the-middle attack**). On the user PC, the certificate must be considered a trusted one; otherwise, it will not be able to connect to an HTTPS resource. All HTTPS resources, which use personal certificates for identification (e.g., different client banks), must be excluded from interception.

Method 1:

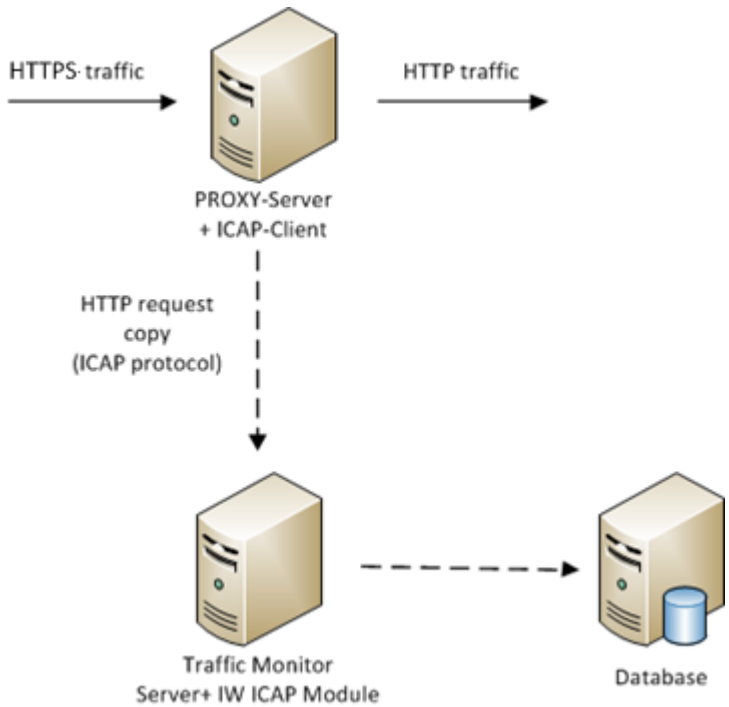
Corporate proxy-server carries out certificate replacement and HTTPS traffic parsing. After parsing HTTPS traffic, the proxy server sends traffic as HTTP to Traffic Monitor Server via ICAP protocol.

The following proxy servers support this option:

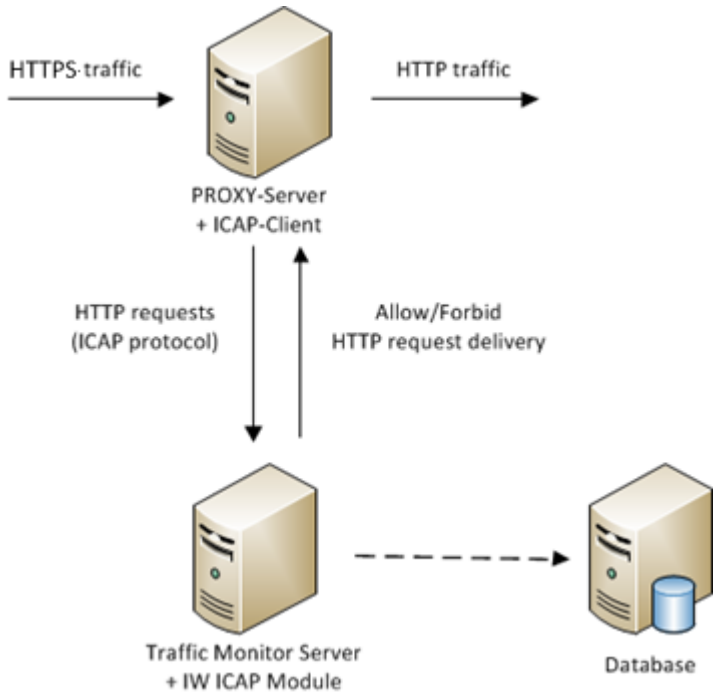
- SQUID (given that the package has been compiled including ICAP support)
- Zimbra Proxy (as a part of Zimbra ZCS)

> **Note:**
>
> To work with other proxy servers, maintaining ICAP, is possible, but requires preliminary compatibility test.

Copy



Blocking



For more information on this functionality, see *Inline Mode*.

Advantages:

- Entire HTTPS traffic is analyzed.

Shortcomings:

- It is required to take into consideration the needs of all proxy-server users;
- Additional load on a proxy server;
- Need to make some preparations which include creating exclusions list and certificate distribution.

Special case:

Install a free SQUID proxy server on the Traffic Monitor Server.

Advantages:

- It is possible to analyze HTTPS.

Shortcomings:

- SQUID engineer is required to maintain the service.

Method 2:

HTTPS traffic analysis is performed by InfoWatch Device Monitor.

For more information on this functionality see *InfoWatch Traffic Monitor. User Guide.*

Advantages:

- An agent automatically puts its certificate to the trusted ones;
- Analysis of HTTPS traffic in company environment can be introduced gradually.

Shortcomings:

- Needs of a particular PC user should be taken into consideration.

ICQ Traffic Interception Scheme

Intercepting the copy of ICQ traffic (OSCAR protocol) which passes through equipment with SPAN technology support. ICQ traffic interception is possible over HTTP.

> **Warning!**
>
> Interception and analysis of encrypted ICQ traffic (including traffic transferred via SSL protocol) is not supported.

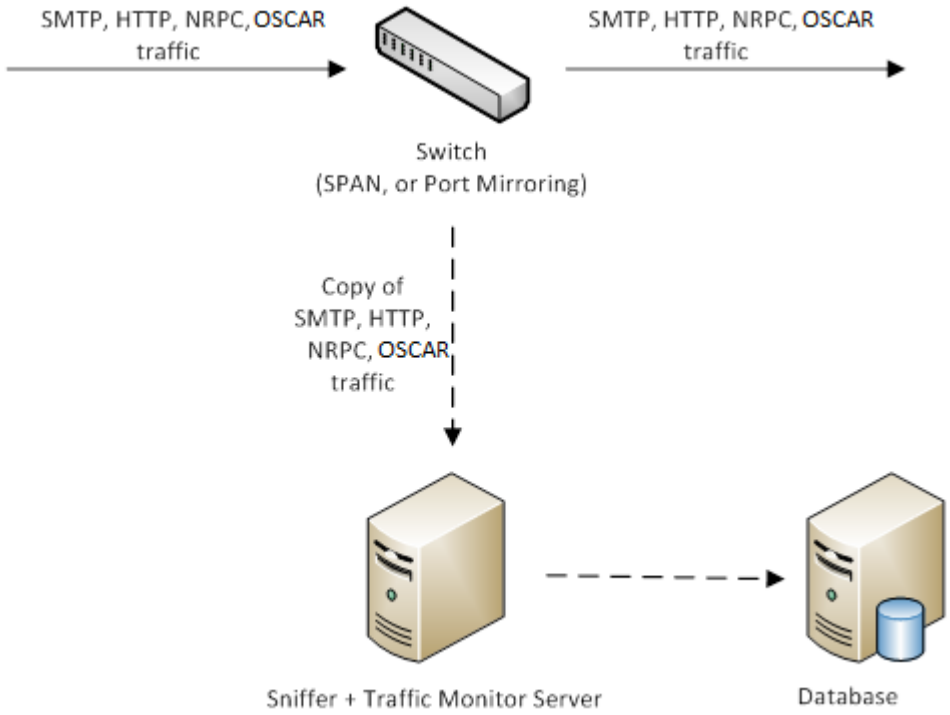
> **Note:**
>
> Direct file transfer is not supported. OSCAR protocol assumes that Traffic Monitor gets the file hosting service reference, from which the file can be loaded.

Traffic is intercepted from a managed switch by using SPAN Port, or Port Mirroring technology.

A large number of managed network switches allow duplicating traffic from one or several ports or VLAN to a single port.

In this case customized switch port is connected to Traffic Monitor Server network interface. Server interface is switched to promiscuous mode, which allows accepting all incoming packages. This interface does not require an IP-address.

It is most effective to mirror internal switch port or internal proxy-server interface.



For more information on functionality see *SPAN, or Port Mirroring*.

Advantages:

Allows analyzing traffic safely even in an isolated network segment. System cannot influence the traffic being analyzed regardless of the Server state.

Shortcomings:

Under heavy load on the switch, some packages can be lost, especially, when the traffic is mirrored from two ports to a single port. To prevent the loss of packages, you should use:

- a switch with corresponding functionality;
- an additional network adapter on Traffic Monitor server.

NRPC Traffic Interception Scheme

NRPC traffic copy interception (used in IBM Notes System) which passes through equipment with SPAN technology support.

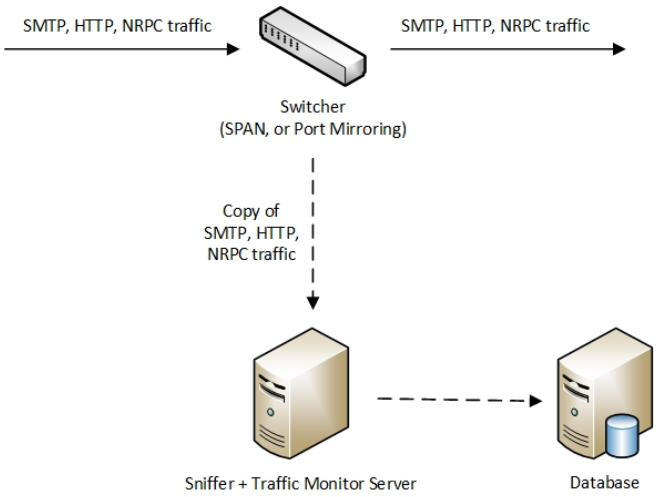
Method 1:

Traffic is intercepted from a managed switch by using SPAN Port, or Port Mirroring technology.

A large number of managed network switches allow duplicating traffic from one or several ports or VLAN to a single port.

In this case customized switch port is connected to Traffic Monitor Server network interface. Server interface is switched to promiscuous mode, which allows accepting all incoming packages. This interface does not require an IP-address.

The most effective is to mirror internal switch port or internal proxy-server interface.



For more information on functionality see *SPAN, or Port Mirroring*.

Advantages:

Allows analyzing traffic safely even in an isolated network segment. System cannot influence the traffic being analyzed regardless of the Server state.

Shortcomings:

Under heavy load on the switch, some packages can be lost, especially, when the traffic is mirrored from two ports to a single port. To prevent the loss of packages, you should use:

- a switch with corresponding functionality;
- an additional network adapter on Traffic Monitor server.

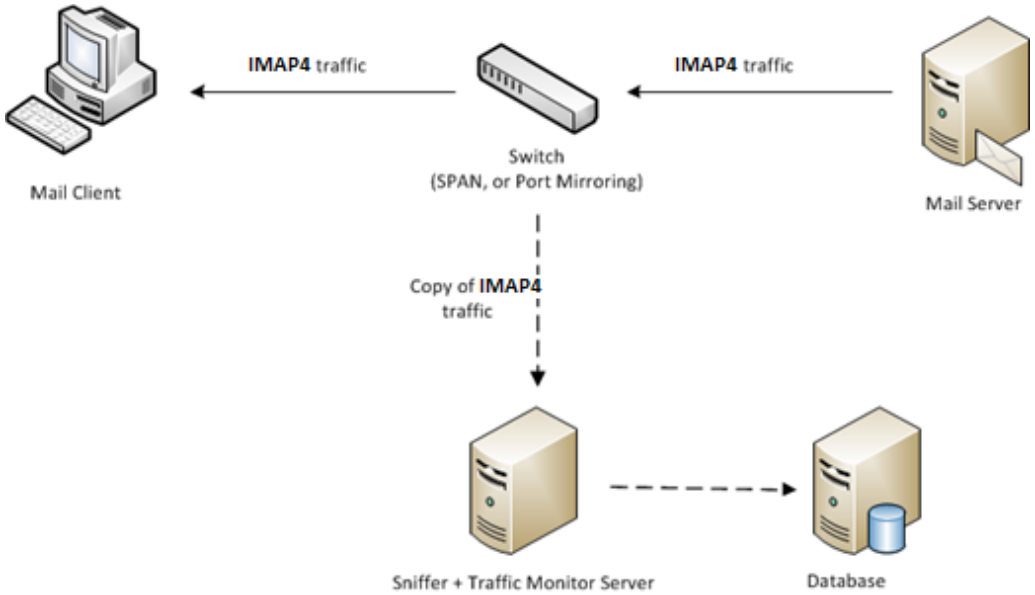
IMAP4 Traffic Interception Scheme

Traffic is intercepted from a managed switch by using SPAN Port, or Port Mirroring technology.

A large number of managed network switches allow duplicating traffic from one or several ports or VLAN to a single port.

In this case customized switch port is connected to Traffic Monitor Server network interface. Server interface is switched to promiscuous mode, which allows accepting all incoming packages. This interface does not require an IP-address.

It is most effective to mirror the internal switch port or internal proxy-server interface.



For more information on functionality see *Configuring Server to Work with SPAN, or Port Mirroring Technology*.

2.1.2 Processing Traffic, Received from Third-party Systems

Traffic Monitor supports number of adapters, which work under the scheme of data transfer from integrable source and storing it in a database:

- **InfoWatch Lotus Domino Adapter**. Find the recommendations for installation and administration in the document *InfoWatch Lotus Adapter. Installation and Configuration Guide*;
- **InfoWatch MS Lync Adapter**. Find the recommendations for installation and administration in the document *InfoWatch MS Lync Adapter. Installation and Configuration Guide*;
- **InfoWatch TMG Adapter**. Find the recommendations for installation and administration in the document *InfoWatch TMG Adapter. Installation and Configuration Guide*;
- **InfoWatch Device Control Adapter**. Find the recommendations for installation and administration in the document *InfoWatch Device Control Adapter. Installation and Configuration Guide*;
- **InfoWatch Cisco UCM Adapter**. Find the recommendations for installation and administration in the document *InfoWatch Cisco UCM Adapter. Installation and Configuration Guide*;
- **InfoWatch Smart Logger II**. Find the recommendations for installation and administration in the document *InfoWatch Smart Logger II. Installation and Configuration Guide*;
- **InfoWatch HEML Adapter**. Find the recommendations for installation and administration in the document *InfoWatch HEML Adapter. A Guide for Installation, Configuration, and Running the Program*;

- **InfoWatch Sample Documents Autoupdate Adapter**. Find the recommendations for installation and administration in the document *InfoWatch Sample Documents Autoupdate Adapter. Setup and Configuration Guide*.

2.2 InfoWatch Traffic Monitor Subsystems (Red Hat Enterprise Linux)

InfoWatch Traffic Monitor subsystem	Purpose
Traffic Intercepting subsystem	Intercepts traffic (objects or their copies) and transfers it for processing. Consisting of the following modules: • Sniffer module • ICAP module • Device Monitor module • Mail (SMTP) interception module
Processing subsystem	Extracts significant information and attachments from intercepted objects, determines attachment formats, and transfers extracted texts to the Analysis subsystem. **Note:** together with an object, its XML context is created - a text file including object content and information about the object.
Analysis subsystem	Analysis of text data extracted from intercepted objects (texts of emails, messages, queries as well as text, extracted from attachments). Includes the following technologies: • Categories and terms; • Text objects; • Sample documents; • Vector images; • Blanks; • Stamps; • DB unloadings; • Graphical objects; • Autolinguist.
Policy Application subsystem	Based on results of Analysis and Processing subsystems operations, the Policy Application subsystem makes a decision whether the intercepted object violates the security policy or not. It also associates the sender/recipient data with records on employees and workstations. Consists of the following modules: • Active Directory, Samba DC and Domino Directory Integration module • Decision-Making module

InfoWatch Traffic Monitor subsystem	Purpose
Storage subsystem	A database which stores data on intercepted objects, results of analysis and applying policies. It also enables viewing this data by means of queries in the Management Console. Includes the following modules: • Remote Database Management module • Uploading Objects to Database module • System Settings Storage module • Object Storage module **Note:** before objects are saved to the database, they are converted from one internal format (XML) to another internal one.
Monitoring subsystem	Allows you to remotely monitor the state of the servers where InfoWatch Traffic Monitor components and their services are installed. Also, performs general management operations. The administration works with the subsystem via the web interface.
Audit subsystem	Makes it possible to customize search filters (by persons, actions, objects, dates), view brief information on events and violations based on selected filters, and define the period of storing events in the System. The administration works with the subsystem via the web interface.
Management Console subsystem	Enables GUI operation. The user interface is needed to administrate, configure, and operate Traffic Monitor. Includes the following modules: • Monitoring module • Control module • Configuration module

2.3 Licensing

After the System has been installed, the license should be installed accordingly. For this purpose, request for license key file (see the document *InfoWatch Traffic Monitor. User Guide. Requesting License*).

The license is intended to specify the duration, number of users, number of interception modules and analysis modules, as well as the ability to integrate with external systems.

A commercial license key is a file in LIC format.

When you install and use the license key, consider the following:

- If the license expires, interceptors stop working. To resume the System operation, the new license should be installed.
- If you want to change settings of traffic interception according to a new deployment scheme, replace the license key considering new interceptors.

If you completely reinstall the operating system or InfoWatch Traffic Monitor, you will need to install the license anew. Therefore, we recommend saving the license key file to some data storage.

Warning!

About license validity check and license management, you may find in the document *InfoWatch Traffic Monitor. User Guide. Managing Licenses*.

Modules used in the product are listed below.

Interception modules designed by InfoWatch		
Subsystem	Event type	Protocol
TM	E-Email	POP3 IMAP SMTP NRPC
	Web message	HTTP HTTPS
	Web email	HTTP HTTPS
	ICQ	OSCAR
	Data Discovery	-
DM	Email	POP3 IMAP Outlook SMTP
	Web message	HTTP HTTPS
	Web email	HTTP HTTPS
	Print	-
	Skype	SKYPE

Interception modules designed by InfoWatch		
Subsystem	Event type	Protocol
	XMPP	XMPP
	Mail.Ru Agent	MMP
	FTP	FTP
	Clipboard	-
	Screenshots	-
	Removable devices	-
	Network resources	-
	Terminal session	-
	Cloud storage	HTTPS
	Telegram	-
	Keyboard input	-
	VK (VKontakte)	HTTPS
	Facebook	HTTPS
	WhatsApp	-
Adapters	MS Lync	SIP
	Email (Lotus)	NRPC
	Web message (ICAP)	ICAP
	Web email (ICAP)	ICAP

External interception modules	
Event type	Protocol
Email	POP3 MAPI SMTP/ESMTP IMAP NRPC
Web email	HTTP HTTPS ICAP
ICQ	OSCAR
Mail.ru agent	MMP
Skype	Skype
XMPP	XMPP
MS Lync	SIP
Web message	HTTP HTTPS ICAP
FTP	FTP
Removable devices	-
Print	-
SmartLogger	XMPP

Analysis modules:

1. Linguistic analysis
2. Sample documents detector
3. Vector images detector
4. Text objects detector
5. DB unloadings detector
6. Form detector
7. Graphical analysis
8. Stamps detector
9. Autolinguist

Additional possibilities:

1. Modules of sample unloadings update;
2. Export to external analysis systems.

> **Note:**
>
> The technology of blocking leakage channels on workstations based on analysis results is not licensed.

3 System Installation Types

The deployment scheme is chosen based on estimated hardware workload and implementation goals (for details, see the document *InfoWatch Traffic Monitor. Installation Guide. System Deployment Schemes and Installation Types*). The following installation modes are supported:

- *All-in-one Standard* - the base type of installation, when all System components are installed on the same computer together with PostgreSQL DBMS. For details, see the document *InfoWatch Traffic Monitor. Installation Guide. All-in-one Installation Mode of TM Enterprise and TM Standard*;
- *All-in-one Enterprise* - System installation type with advanced features including PostgreSQL and Oracle DBMS, scalability, fine control of workstations. For details, see the document *InfoWatch Traffic Monitor. Installation Guide. All-in-one Installation Mode of TM Enterprise and TM Standard*;
- *Distributed Installation of TM Enterprise (Traffic Monitor Server + Database)* - System installation type for operation under heavy workload and with a large amount of data. For details, see the document *InfoWatch Traffic Monitor. Installation Guide. Distributed Installation of TM Enterprise*.

Important!

If the System is installed on different servers (and when building a server cluster), pay attention to the following:

- you should configure network parameters of Sphinx searching engine (see *Configuring Sphinx for Distributed Installation*)
- some server processes (**iw_adlibitum, iw_bookworm, iw_deliver, iw_licensed, iw_indexer, iw_is, iw_image_autoling, iw_text_autoling**) and user console processes (**iw_kicker, iw_configerator**) should be run on the cluster in a single copy (see *List of Traffic Monitor Server End Processes*)

Depending on the purchased license, each installation type may include installation of the following interceptors:

- InfoWatch Device Monitor which is intended to configure security schemes, computer monitoring system, access to the company computers, etc. (see the document *InfoWatch Device Monitor. Installation, Configuration and Administration Guide*).
- Adapters - interception modules intended for integration with third-party systems.

4 Configuring System After Installation

After installing the System, configure the following settings for proper System operation:

- Configuring Time Synchronization - enables automatic time synchronization on servers;
- Configuring Sphinx for Distributed Installation - settings required for distributed installation;
- Configuring Traffic Interception - specific settings for different types of intercepted traffic;

- General Settings – the settings order to get traffic from the managed switch using SPAN Port technology, traffic interception, sent via ICAP protocol and system settings to switch the interceptors to "in-line" mode capably for traffic block;

- Process autorun - the list of system processes and description when and how to enable/disable autorun for these processes;
- Configuring OCR Extractors - the order of installing packages required for text recognition in intercepted events;
- Configuring Notifications for Users and Employees - mandatory settings to support mail notifications, sent as a result of triggering policy rules (for details, see the document *InfoWatch Traffic Monitor. User Guide*);
- Limiting the number of events that can be found - changing the maximum number of events displayed in the Management Console;
- Configuring InfoWatch Device Monitor - configuring separate modules of Device Monitor Server and editing the logging settings (for details, see the document *InfoWatch Device Monitor. Installation, Configuration and Administration Guide*);
- Configuring Data Collection in a Branch Network - the settings of configuration files services, responsible for collecting data in a branch network;
- Service-to-service cooperation setup (Consul service) - the settings of configuration files services, managing The System processes;
- Detecting Workstations to Install the Agents on - scanning the network in order to detect workstations outside Active Directory (for details, see the document *InfoWatch Device Monitor. User Guide*).

We recommend:

- changing passwords pre-configured in the System, as information security rules require (for details, see *Changing the Preset Password*);
- configuring protection against guessing passwords during authorization in Traffic Monitor and Device Monitor consoles (for details, see *Configuring protection against password guessing*);
- configuring a firewall to prevent unauthorized access to the System components (for details, see *Secure Deployment Recommendations*).

4.1 Changing the Preset Password

For accounts in the System, a standard password is preset – xxXX1234 (for more details, see the article *InfoWatch Traffic Monitor. Installation Guide. Pre-installed Server Parameters*). During the operation of the System, it must be replaced, following information security requirements. Stable and secure operation of the System requires storing passwords in a secure place, inaccessible to others. The standard password is changed in the System configuration files. To replace the preset password with a new one:

Stop process Traffic Monitor:

1. Stop process Traffic Monitor:

```
iwtm stop
```

2. Run command:

```
egrep -lir --include=\*.{cfg,conf,pgpass} --exclude-dir=\*{etc_conf_backup,config_merge}\*
'xxXX1234' /  | xargs -l sed -i -e 's/xxXX1234/<new_pass>/g'
```

where `<new_pass>` is the new password.

3. Change the password to connect to the database. For this:

For PostgreSQL:

a. Connect to PostgreSQL DBMS:

```
psql postgres iwtm
```

b. Get a list of users:

```
\du
```

c. Change user passwords:

```
alter user <iw_user> with password <'new_pass'>
```

where `<iw_user>` is the selected database user, `<'new_pass'>` is the new password for this user.

d. Disconnect from the PostgreSQL DBMS:

```
\q
```

For Oracle:

a. Connect to the Oracle DBMS:

```
sqlplus iwtm@iwtm
```

b. Enter the preset password.

c. Get a list of users:

```
select * from dba_users
```

d. Change user passwords:

```
ALTER USER user IDENTIFIED BY new_pass
```

where `user` is the selected database user, `new_pass` is the new password for this user.

e. Disconnect from the Oracle DBMS:

```
exit
```

4. Start process Traffic Monitor:

```
iwtm start
```

Requirements for passwords

The password may include the following symbols:

- Capital Latin characters (A-Z);
- Lowercase Latin characters (a-z);
- Arabic numerals (0-9);
- Special symbols: "@", "%", ":", "^", "-", "_", "+", "/", ".", ",".

4.2 Configuring protection against password guessing

To protect the System, the following feature is enabled: in the case of repeated unsuccessful authorization attempts, user login is blocked. Protection against password guessing is configured in the database.

By default, the protection is disabled.

Protection for InfoWatch Traffic Monitor and InfoWatch Device Monitor is configured separately.

To enable protection for Traffic Monitor server:

1. Connect directly to the database which is used by the Traffic Monitor server.
2. Find the table `SETTING`.
3. Find 3 parameters responsible for protection configuration:
   - **`web_logon_attempts_count`** - the number of unsuccessful login attempts allowed;
   - **`web_logon_blocking_period`** - the time for which the login will be blocked, in seconds;
   - **`web_logon_attempts_period`** - the period of unsuccessful login attempts, in seconds.
4. Set the values for all three parameters in accordance with security requirements.

> **Warning!**
>
> In case any parameter's value is incorrect or equal to 0, the protection is to be disabled.

5. To apply the settings, save changes in the database. You need no restart the server as protection settings change dynamically.

If login is blocked, a new event is created in the audit log (the **Log** button in the Navigation pane) where the unlocking time is specified.

When the blocked user tries to log into the System again, he/she will be notified that the number of failed login attempts has been exceeded.

An example of protection settings for Traffic Monitor:

- `web_logon_attempts_count - 5`
- `web_logon_blocking_period - 300`
- `web_logon_attempts_period - 120`

How these settings work: if during 2 minutes (120 seconds), the user makes 5 unsuccessful login attempts, then login for this user is blocked for 5 minutes (300 seconds). The countdown period of unsuccessful attempts starts from the first attempt. If during 120 seconds the user makes only 4 unsuccessful login attempts, then the next attempt will relate to a new period.

4.3 Preliminary Settings

Upon System installation the following settings must be performed:

- Configuring Time Synchronization;
- Configuring Sphinx for Distributed Installation.

4.3.1 Configuring Time Synchronization

1. Set the system time using `date` command. For example, to set 12 August 2013 13:30 run the command with following parameters:
   `date 12081330`
2. Copy the system time to configure hardware clock with the command:
   `hwclock --systohc`
3. Check if the system and hardware clock are configured properly:
   `hwclock ; date`
4. The time should be almost the same.
   Stop `chronyd` service:
   `systemctl stop chronyd`
5. Define the time synchronization server. You can use any exact time service which uses ntp protocol and is available from your network. It can be both network equipment and Windows domain controllers (Active Directory server).
   To check if the server supports NTP, use `chronyd -q 'server <IP> iburst'` command (where IP is the address of server to be checked), for example:
   ```
   chronyd -q 'server 10.10.0.98 iburst'
   2024-09-11T10:30:17Z chronyd version 4.3 starting (+CMDMON +NTP +REFCLOCK
   +RTC +PRIVDROP +SCFILTER +SIGND +ASYNCDNS +NTS +SECHASH +IPV6 -DEBUG)
   2024-09-11T10:30:17Z Initial frequency -19.107 ppm
   2024-09-11T10:30:22Z System clock wrong by 8551737.490761 seconds (step)
   2024-12-19T09:59:19Z chronyd exiting
   ```
6. Configure the synchronization specifying NTP synchronization server in **/etc/chrony.conf** file. Add entry (specify IP address of your NTP server):
   `server 10.10.0.98 iburst`
7. Run `chrony` service:
   `systemctl start chronyd`
8. Check the current state of `chrony` service with the command:
   `systemctl status chronyd`
9. Enable automatic time synchronization:
   `chronyc -N sources`

```
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================
^* 10.10.0.98                    4   6    37    19    -14us[ -102us] +/-   39ms
```

4.3.2 Configuring Sphinx for Distributed Installation

Full-text query search in Traffic Monitor is implemented by means of Sphinx search engine. **Sphinx** service is installed in **All-in-one** when using Indexer template. For information on installation, see the document *InfoWatch Traffic Monitor. Installation Guide. System Deployment Schemes and Installation Types*).

A distributed installation is an installation scheme when Traffic Monitor components are installed on different servers.

> **Note:**
>
> If you use several servers, specify the server IP address where **sphinx** service is running (a server with Indexer template). IP address is specified in `hostname` parameter within `search` section of **web.conf** configuration file on the server with Web console template used.

4.4 General Settings

The chapter contains information on configuring the System and its components:

- Configuring Server to Work with SPAN, or Port Mirroring Technology;
- Intercepting Traffic, Transmitted via ICAP Protocol;
- Configuring Licensing Server (in case of Cluster).

4.4.1 Configuring Server to Work with SPAN, or Port Mirroring Technology

Traffic is being intercepted from managed switch via SPAN Port technology, or Port Mirroring.

There are different possible deployment schemas:

Variant 1. Traffic copy interception and analysis are being performed on one computer (**All-in-one kickstart** key). Cluster is not created:

1. Configure Server to work as Sniffer (see *Configuring Traffic Monitor Server as Sniffer*, variant 1).
2. Configure traffic reception parameters from Sniffer (see *Configuring Traffic Monitor Server to receive Copy of Traffic from Sniffer*).

Variant 2. Traffic copy interception and analysis are being performed on different computers, clusterization is not used:

1. On one computer (**Interceptors** template key):
    a. Configure Server to work as Sniffer (see *Configuring Traffic Monitor Server as Sniffer*, variant 2).
2. On the other computer (**All-in-one kickstart** key):
    a. Configure traffic reception parameters from Sniffer (see *Configuring Traffic Monitor Server to receive Copy of Traffic from Sniffer*).

Variant 3. Traffic copy interception and analysis are being performed on different computers. To improve the performance, create a cluster of two Traffic Monitor server instances:

1. Install and configure DBMS server (**Database** and **Indexer** templates key).
2. On the computer, where Sniffer will be running (**Interceptors** template key):
    a. Configure it to work as Sniffer (see *Configuring Traffic Monitor Server as Sniffer*, variant 2).
3. On the computer, where Traffic Monitor cluster example will be running (**All-in-one** key):

a. Configure traffic reception parameters from Sniffer (see *Configuring Traffic Monitor Server to receive Copy of Traffic from Sniffer*).
b. Specify clusterization parameters (see *Creating Traffic Monitor cluster*).
c. Configure licensing server (see *Configuring Licensing Server (in case of Cluster)*).

Repeat i.3 to add one more Traffic Monitor Server duplicate copy to cluster.

Configuring Traffic Monitor Server as Sniffer

For Traffic Monitor Server to function as Sniffer, one needs to perform configuration file setup.

In **sniffer.conf** configuration file specify required parameters in sections for each interceptor (sections `http`, `icq` and `smtp`):

Parameter	Description
`Interface`	The network interface used to transfer data from the switch
`OpenInSec` (subsection `Timeouts`)	Time to keep the connection alive (in seconds) if no packages are transmitted. The recommended value – `600`.
`LiveInSec` (subsection `Timeouts`)	Timeout (in seconds) when the status of the connection must change to *ESTABLISHED*. If the status is not changed, the connection will be automatically terminated. Recommended value – `86400`.
`CloseInSec` (subsection `Timeouts`)	Time to keep the connection alive (in seconds) when the status is changed to *TIME_WAIT*. The recommended value – `300`
`TailInSec` (subsection `Timeouts`)	Wait time for sent packages (in seconds) after the connection has been terminated
`QueueMemorySizeInBytes`	The size of memory (in bytes), which is used to receive packages. When the channel utilization is high, you can increase this value. The range of values: from 1 to 500 MB. The recommended value – `16777216`

1. Stop server processes of Traffic Monitor:

```
iwtm stop
```

2. Configure process autostart:
   - Variant 1. Traffic copy interception and analysis are performed on the same computer. Traffic Monitor Server will receive traffic copy from a switch (Sniffer function), analyze received data and upload it to the database (Traffic Monitor Server function). Enable **iw_sniffer** process autostart:

     ```
     iwtm enable iw_sniffer
     ```

   - Variant 2: Traffic copy interception and analysis are performed on different computers. Configured Traffic Monitor Server example will act only as Sniffer (traffic interception).

a. Enable **iw_sniffer** process autostart:

```
iwtm enable iw_sniffer
```

b. Disable other processes autostart.

> **Note:**
>
> For more information on enabling and disabling autostart, see *Enabling and Disabling Process Autorun*.

3. In **/opt/iw/tm5/etc/sniffer.conf** file, section `iscp`, configure traffic interception parameters:
   - `ListenHost` – IP-address on which the **iw_proxy** process receives incoming connections from **iw_sniffer**;
   - `ListenPort` – Port via which the **iw_proxy** process receives incoming connections from **iw_sniffer** (4301 by default).
4. Start Traffic Monitor processes:

```
iwtm start
```

Configuring Traffic Monitor Server to Receive Copy of Traffic from Sniffer

Configuration is performed on the computer where Traffic Monitor Server is installed. Configuring these settings is required if Sniffer and Traffic Monitor Server are installed on the same computer.

Go to `/opt/iw/tm5/etc` directory and configure the following settings:

1. Enable autostart for required traffic interceptors (setup for SMTP, HTTP(S), ICQ traffic:

```
iwtm enable iw_proxy_http iw_proxy_smtp iw_proxy_icq
```

   For NRPC traffic interceptor autostart configuration description, see *SPAN, or Port Mirroring*.
2. Make sure **proxy.conf** configuration file has `false` meaning for `SkipNegotiate` parameter.
3. In case you need to intercept ICQ traffic, which is being transmitted via proxy-server (over HTTP), set `true` value for `IcqFilter` parameter in **proxy.conf** configuration file.
4. In **proxy.conf** configuration file, in `Iscp` part of sections `Smtp`, `Icq`, `Http` create values of `Host` and `Port` parameters, which are used to call the **iw_sniffer** process.

> **Note:**
>
> To configure interaction with the computer used to run the **iw_sniffer** process, which has 10.20.30.40 IP address and port **1234**, `Iscp` section for ICQ interceptor will look as follows:

```
"iscp": {
"ListenArea": "icq",
"Host": 10.20.30.40,
```

```
"Port": 1234
}
```

Creating Traffic Monitor Cluster

If the traffic copy is intercepted by Sniffer, then you can create clusters of Traffic Monitor servers in order to increase the System performance.

> **Warning!**
>
> If a cluster is used, then the speed of a network connection between Sniffer and Traffic Monitor server should be twice as large as the speed at which traffic arrives at Sniffer.

To create a cluster of Traffic Monitor servers on each TM server instance that will be included in the cluster:

1. In the **sniffer.conf** configuration file, make sure that the `Usedareas` section looks like this:

   ```
   "UsedAreas": [
   "capstack",
   "http",
   "icq",
   "smtp"
   ]
   ```

   By default, this parameter is specified.
2. In the **proxy.conf** configuration file, in `Iscp` part of `Smtp`, `Icq`, and `Http` sections, specify the values of `Host` and `Port,` on which the **iw_sniffer** process is running.
   For example, to set up interaction with a computer which has an IP address of 10.20.30.40 and port 4301 (by default), the `Iscp` part for ICQ interceptor will look like this:

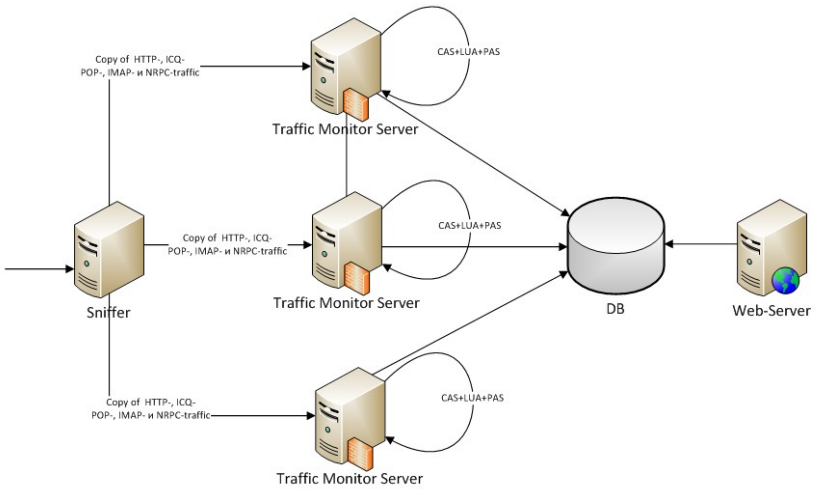
   ```
   "iscp": {
   "ListenArea": "icq",
   "Host": 10.20.30.40,
   "Port": 4301
   }
   ```

3. In the **sniffer.conf** configuration file, specify the value of the `Balancer` parameter for `Smtp`, `Icq`, and `Http` sections. Possible values:
   - `Easy` – for each new TCP connection, an instance of Traffic Monitor Server with **iw_proxy** is chosen. In this case, all Traffic Monitor Server instances are used one in a turn.
   - `IpPref` – the traffic is distributed between several instances of Traffic Monitor server with **iw_proxy** based on an IP address of the client. In this case, all data with the same IP address of the client is transmitted to the same instance of Traffic Monitor server.

   By default, the `IpPref` parameter is used.

Example. Creating a cluster of several (up to three) Traffic Monitor server instances

The System intercepts a copy of SMTP, ICQ and HTTP traffic via Sniffer. To increase the performance, you can create a cluster of several (three in this case) Traffic Monitor Server instances. With the `Balancer` parameter, you can choose the optimal type of load on Traffic Monitor servers. This option allows you to avoid delays in processing data and losing data packages. The servers analyze traffic, then the data is saved to the database for further usage.



4.4.2 Intercepting Traffic, Transmitted via ICAP Protocol

A proxy server, supporting ICAP, intercepts HTTP traffic and delivers it to the Traffic Monitor server for analysis and uploading to the database.

In this scheme, automatically installed modules are used on the Traffic Monitor server:

- **iw_icap_buf**, which receives large data in parts, combines these parts and send data to **iw_icap** for analysis;
- **iw_icap**, which processes intercepted data received through a proxy server or data processed by **iw_icap_buf** module.

The scheme supports user data processing with the following methods of authentication: NTLM, LDAP, Basic, Digest.

Warning!

Traffic Monitor Server and proxy with ICAP client must be located in the same subnetwork.

The web-browser of each workstation must be configured to use a SQUID proxy server for HTTP(S) traffic. For more information on configuring the settings, see the documentation for your web browser.

Important!

The **iw_icap_buf** supports buffering of file uploads to file exchangers: filetrance.tander.ru, Yandex.Disk (disk.yandex.ru), Mail.ru Cloud (cloud.mail.ru), Google Drive (drive.google.com), BOX (box.com), DropBox (www.dropbox.com), OneDrive (onedrive.live.com). When uploading files to another file storage, buffering is not supported.

Depending on transmission type (**Copy** or **Block**), the following variants of HTTP traffic are possible:

- **Copy** mode – a copy of HTTP traffic is transmitted to TM Server for analysis; traffic is transmitted via proxy-server to the next chain link, depending on corporate infrastructure.
- **Block** mode – HTTP traffic is analyzed by the Traffic Monitor system. Depending on the fact if there is a violation or not, the following routes of traffic are possible:
  - In case there is no violation, a message which allows traffic transmission is returned to a proxy server. From the proxy-server, traffic is forwarded to the next link in a chain, depending on corporate infrastructure;
  - In case there is a violation, a message which prohibits traffic transmission is returned to a proxy-server. Traffic is blocked and the user, who sent the traffic, gets a warning message.

> **Note:**
>
> The text of the error message is contained in the **error.html** file which is located in the `/opt/iw/tm5/etc` directory.

In this chapter:

- Configuring ICAP;
- Recommendations on SQUID Configuration;
- Disabling ICAP.

Configuring ICAP

The IW ICAP module is automatically installed with Traffic Monitor server (**All-in-one** installation or **Interceptors** template options). Autostart of the **iw_icap** process on TM Server is enabled by default.

HTTP and HTTPS traffic is intercepted with the same System resources, but the configuration of external mechanisms configuration differs (see below).

To enable interception of HTTP(S) traffic transferred via ICAP, you need to configure parameters of intercepting HTTP and HTTPS requests on a proxy server.

> **Warning**
>
> On each workstation, configure the browser so that HTTP(S) traffic should pass through the SQUID proxy server. For more information on how to configure this, see the documentation for your browser.

The following ICAP configuration options are possible:

1. For the traffic to be transmitted in "Copy" mode:
   a. In the **luaengined.conf** configuration file, which is located in the `/opt/iw/tm5/etc` directory, in the `Icap` property of the `TransportModes` section, specify a valid FQDN name of the TM server (for example, `tm-server1.local`). The value of the mode should be: `Copy`. You can specify multiple pairs of values:
   `"Icap": [{"Server": "tm-server1.local", "Mode": "Copy"}, {"Server":"tm-server2.local", "Mode": "Copy"}]`
   b. Save changes in the **luaengined.conf** file.
2. For the traffic to be transmitted irrespective of the verdict:
   a. In the code specified above, set the mode value to `Transparent`.

b. Save changes in the **luaengined.conf** file.

3. For the traffic to be transmitted in the "Block" mode:
   a. In the code specified above, set the move value to `Normal`.
   b. Save changes in the **luaengined.conf** file.
4. In the **icap.conf** configuration file, which is located in the `/opt/iw/tm5/etc` directory, set the value for `AsyncCheck` parameter of the `Icap` section as follows:
   - `true` – the "Copy" mode
   - `false` – the "Block" mode
5. Save the configuration file.
6. Restart the Traffic Monitor server:
   `iwtm restart`

Configure parameters for intercepting HTTP and HTTPS requests on a proxy server.

> **Note:**
>
> This setting depends on the modification of the proxy server. You can find actual information in the documentation for your proxy server.
>
> See *Recommendations on SQUID Configuration*.

Recommendations on SQUID Configuration

Documentation contains information on configuring SQUID proxy server for MS AD user authentication, SSL interception (ssl-bump) and traffic verification via ICAP.

In this chapter:

- SQUID Installation;
- Configuring ICAP;
- Configuring SSL Interception and Disclosure;
- Configuring User Authentication.

SQUID Installation

Before installing SQUID proxy server, download an RPM package or an archive (that should be then unpacked) from a repository or from CD/DVD. In our example we use the `squid-3.4.0.2-2.el6.x86_64.rpm` RPM package.

To install an RPM package of SQUID proxy server, log into the System under `root` and enter the following command:

```
rpm -i squid-3.4.0.2-2.el6.x86_64.rpm
```

If the package is installed successfully, the following information will be displayed:

```
Preparing...        ##################  [100%]
1:squid             ##################  [100%]
```

To access the configuration file, enter:

```
cd /etc/squid/
mcedit squid.conf
```

Configuring ICAP

To configure ICAP, do the following:

1. To the **squid.conf** configuration file, add lines for processing ICAP:

```
icap_enable on
icap_service service_req reqmod_precache bypass=1 icap://10.60.7.7:1344/request
adaptation_access service_req allow all
icap_send_client_ip on
icap_send_client_username on
```

2. To configure working on ICAP **UserGate Web Filter** server, in the `squid.conf` configuration file, add the following:

```
icap_enable on
icap_preview_enable on
icap_preview_size 4096
icap_send_client_ip on
icap_service service_req reqmod_precache bypass=0 icap://10.0.3.10:1344/request
adaptation_access service_req allow all
icap_service service_resp respmod_precache bypass=0 icap://10.0.3.10:1344/response
adaptation_access service_resp allow all http_port 8080 transparent
```

   where `10.0.3.10` is the **UserGate Web Filter** server address.

   Important!

   UserGate Web Filter server does not support `icap_preview_enable off` mode. Working with HTTPS traffic is not supported at the moment.

3. Restart Squid:

```
sudo service squid3 restart
```

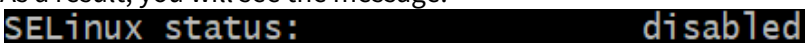
Configuring SSL Interception and Disclosure

SSL-Bump mode is being used for HTTPS sessions encrypted content interception. SQUID performs SSL-connection with server and receives its certificate as first intercepted HTTP request is transmitted. After that SQUID uses host name from real, received from server certificate and creates fictitious certificate whereby simulates requested server in interaction with client, wherein proceeding on using SSL connection. Except for HTTPS connections, specified scheme can be used to intercept major HTTP requests.

To configure SSL-Bump mode, perform the following actions:

1. Disable the SELinux access control system. For this:
   a. Run the command:
      `sudo sed -i 's/^SELINUX=.*/SELINUX=disabled/g' /etc/selinux/config`
   b. Restart the server using the command:
      `reboot`
   c. Check status of the SELinux system using the command:
      `sestatus`

As a result, you will see the message:



2. Create a catalogue to store SQUID certificates:
   `mkdir /etc/squid/ssl_cert`
3. Generate a root certificate for browsers of workstations whose traffic will be intercepted:
   `openssl req -new -newkey rsa:2048 -days <number of days> -nodes -x509 -keyout /etc/squid/ssl_cert/IW.key -out /etc/squid/ssl_cert/IW.crt`
   `openssl x509 -in /etc/squid/ssl_cert/IW.crt -outform DER -out /etc/squid/ssl_cert/IW.der`
4. Copy the file with extention **.der** to the workstations and install it as a trusted certificate to computer storage.
5. Configure browsers of the workstations to use new proxy server SQUID.
6. Generate a parameter file for the Diffie-Hellman algorithm:
   `openssl dhparam -outform PEM -out /etc/squid/ssl_cert/IW_dhparam.pem 2048`
7. Make the **squid** user an owner of the `/etc/squid/ssl_cert` directory and, to ensure security, change file permissions:
   `chown -R squid:squid /etc/squid/ssl_cert && chmod 400 /etc/squid/ssl_cert/IW*`
8. Identify the SQUID version used on your server. For this, run the command:
   `squid -v`
   Information about used version will be displayed in the following format:
   `Squid Cache: Version <version>`
9. Stop SQUID:
   `systemctl stop squid`
10. If you use SQUID proxy server version 3, perform the following actions:
    a. Create a database for substitute certificates and make the **squid** user its owner:
       `mkdir -p /var/lib/squid`
       `rm -rf /var/lib/squid/ssl_db`
       `/usr/lib64/squid/ssl_crtd -c -s /var/lib/squid/ssl_db`
       `chown -R squid:squid /var/lib/squid`
    b. In the `/etc/squid/squid.conf` configuration file, comment out the `http_port` setting and add the following certificate generation parameters:
       `#http_port 3128`
       `http_port 3128 tcpkeepalive=60,30,3 ssl-bump generate-host-certificates=on dynamic_cert_mem_cache_size=20MB cert=/etc/squid/ssl_cert/IW.crt key=/etc/squid/ssl_cert/IW.key cipher=HIGH:MEDIUM:!LOW:!RC4:!SEED:!IDEA:!3DES:!MD5:!EXP:!PSK:!DSS options=NO_TLSv1,NO_SSLv3,NO_SSLv2,SINGLE_DH_USE,SINGLE_ECDH_USE tls-dh=prime256v1:/etc/squid/ssl_cert/IW_dhparam.pem`
       `sslproxy_flags DONT_VERIFY_PEER`
       `sslproxy_cert_error allow all`
       `always_direct allow all`
       `ssl_bump client-first all`
       `ssl_bump server-first all`
       `ssl_bump none all`
       `sslcrtd_program /usr/lib64/squid/ssl_crtd -s /var/lib/squid/ssl_db -M 20MB`
11. If you use SQUID proxy server version 4 or later, perform the following actions:
    a. Create a database for substitute certificates and make the **squid** user its owner:
       `mkdir -p /var/lib/squid`
       `rm -rf /var/lib/squid/ssl_db`

```
/usr/lib64/squid/security_file_certgen -c -s /var/lib/squid/ssl_db -M
20MB
chown -R squid:squid /var/lib/squid
```

b. In the `/etc/squid/squid.conf` configuration file:

i. Add the following lines before the first `http_access` rule:

```
acl intermediate_fetching transaction_initiator certificate-
fetching
http_access allow intermediate_fetching
```

ii. Comment out the `http_port` setting and add the following certificate generation parameters:

```
#http_port 3128
http_port 3128 tcpkeepalive=60,30,3 ssl-bump generate-host-
certificates=on dynamic_cert_mem_cache_size=20MB tls-cert=/etc/
squid/ssl_cert/IW.crt tls-key=/etc/squid/ssl_cert/
IW.key cipher=HIGH:MEDIUM:!LOW:!RC4:!SEED:!IDEA:!3DES:!MD5:!EXP:!
PSK:!DSS
options=NO_TLSv1,NO_SSLv3,SINGLE_DH_USE,SINGLE_ECDH_USE tls-
dh=prime256v1:/etc/squid/ssl_cert/IW_dhparam.pem
sslproxy_cert_error allow all
ssl_bump stare all
sslcrtd_program /usr/lib64/squid/security_file_certgen -s /var/
lib/squid/ssl_db -M 20MB
sslcrtd_children 8
```

12. Start SQUID:

```
systemctl start squid
```

Configuring User Authentication

SQUID authentication service is intended to decode HTTP `Authorization` header and transmit decoded information to a third-party module. User access is being provided in case information is correct. When information is incorrect or missing, SQUID returns 407 error to the client. Thus, all the work on user authentication is being performed by third-party modules. Location of these modules depends on operating system and SQUID version. For example, in Debian OS with SQUID version 3 installed, they are located in catalogue: `/usr/lib/squid3`, in Red Hat Enterprise Linux: `/usr/lib64/squid`. To see what authentication methods are supported by SQUID, third-party modules placement and other parameters, use command `squid -v`:

```
squid -v
Squid Cache: Version 3.4.0.2
configure options: '--host=x86_64-redhat-linux-gnu' '--build=x86_64-redhat-
linux-gnu' '--program-prefix='
'--prefix=/usr' '--exec-prefix=/usr' '--bindir=/usr/bin' '--sbindir=/usr/sbin'
'--sysconfdir=/etc' '--datadir=/usr/share'
'--includedir=/usr/include' '--libdir=/usr/lib64' '--libexecdir=/usr/libexec'
'--sharedstatedir=/var/lib'
'--mandir=/usr/share/man' '--infodir=/usr/share/info' '--exec_prefix=/usr' '--
libexecdir=/usr/lib64/squid'
'--localstatedir=/var' '--datadir=/usr/share/squid' '--sysconfdir=/etc/squid'
'--with-logdir= of $ (localstatedir)/log/squid' '--with-pidfile= of $
(localstatedir)/run/squid.pid'
'--disable-dependency-tracking' '--enable-eui' '--enable-follow-x-forwarded-
for' '--enable-auth'
'--The enable-auth-basic=DB, the LDAP, MSNT, MSNT-multi-domain, NCSA, NIS, PAM,
the POP3, the RADIUS, SASL, SMB, getp
wnam '--enable-auth-ntlm=smb_lm,fake' enable-auth-digest=file, LDAP,
```

```
eDirectory
 '--enable-auth-negotiate=kerberos,wrapper'
 '--enable-external-acl-helpers=wbinfo_group,kerberos_ldap_group,AD_group' '--
enable-cache-digests'
 '--enable-cachemgr-hostname=localhost' '--enable-delay-pools' '--enable-epoll'
'--enable-icap-client'
 '--enable-ident-lookups' '--enable-linux-netfilter' '--enable-removal-
policies=heap,lru' '--enable-snmp'
 '--enable-ssl' '--enable-ssl-crtd' '--enable-storeio=aufs,diskd,ufs,rock' '--
enable-wccpv2' '--enable-esi'
 '--with-aio' '--with-default-user=squid' '--with-filedescriptors=16384' '--
with-dl' '--with-openssl' '--with-pthreads'
 '--disable-arch-native' 'build_alias=x86_64-redhat-linux-gnu'
'host_alias=x86_64-redhat-linux-gnu'
 'CFLAGS= is O2 -g' 'CXXFLAGS= is O2 -g -fPIC'
 'PKG_CONFIG_PATH=/usr/lib64/pkgconfig:/usr/share/pkgconfig'
```

Third-party modules interaction and settings description is being performed using `auth_param` parameter. Client will alternately attempt to use authentication schemes in the order they have been set in the `squid.conf` configuration file. Newly configured authentication scheme enters into force after SQUID service restart. To change existing schemes without service restart, use `squid -k reconfigure` command.

To control internet access, create acl, which will use configured authentication module. To do this, acl in `selection_type` field must have `proxy_auth` , `proxy_auth_regex` or `external` value using `%LOGIN` variable. Set access permission in `http_access` parameter for specified acl. Below the format is listed, which specifies the type of used authentication (format of using `auth_param` parameter):

```
auth_param authentication_scheme schem_parameters [parameters_value]
```

To control internet access:

1. In the beginning of the `/etc/squid/squid.conf` configuration file, add the `auth_param` authentication setting according to the scheme:
   ```
   auth_param authentication_scheme scheme_parameters [parameters_value]
   ```
2. Create an access control list that will use the configured authentication module. In the configuration file, the access control list is designated as `acl` . In `selection_type` field, the access control list must have `proxy_auth` , `proxy_auth_regex` or `external` value using `%LOGIN` variable. For example:
   ```
   acl auth proxy_auth REQUIRED
   ```
3. Set access permission in `http_access` parameter for the specified access control list. For example:
   ```
   http_access allow auth
   ```

> **Note:**
>
> For the details about configuring user authentication, see documentation for SQUID proxy server.

Example of configuring Basic authentication:

On the server with installed SQUID, perform the following actions:

1. Add the following lines into the beginning of the `/etc/squid/squid.conf` configuration file:
   ```
   auth_param basic program /usr/lib64/squid/basic_ldap_auth -R -b
   "dc=<second-level domain name>,dc=<top-level domain name>" -D "<user
   name>@<Active Directory domain>" -w "<user password>" -f
   "sAMAccountName=%s" <IP address of the Active Directory domain controller>
   ```

```
auth_param basic children 10
auth_param basic realm Squid proxy-caching web server
auth_param basic casesensitive off
auth_param basic credentialsttl 1 minute
acl auth proxy_auth REQUIRED
http_access allow auth
```

2. To enable logging of events in debug mode, in the `/etc/squid/squid.conf` configuration file add the `-d` parameter to the first string. For example:
   `auth_param basic program /usr/lib64/squid/basic_ldap_auth -R -d -b "dc=<second-level domain name>,dc=<top-level domain name>" -D "<user name>@<Active Directory domain>" -w "<user password>" -f "sAMAccountName=%s" <IP address of the Active Directory domain controller>`
   Debug events will be written to the `/var/log/squid/cache.log` file.
3. Restart SQUID, running the command:
   `service squid restart`

Recommendations on configuring UserGate

As an ICAP-client, UserGate allows you to transmit HTTP/HTTPS traffic, SMTP and POP3 mail traffic to Traffic Monitor ICAP server. To set up interaction between UserGate and ICAP server:

1. Create an ICAP-server:
   i. Go to **Security policies** → **ICAP servers**.
   ii. Click **Add** and create one or more ICAP servers by filling in the fields:

Field	Description
Name	ICAP server name
Description	ICAP server description (optional)
Address	IP address of the ICAP server
Port	ICAP server TCP port, the default value is 1344
Message maximum size	Defines maximum size of a message (in KB) to be transmitted to ICAP server. The size should be greater than 0. **Example:** `35840`
ICAP server accessibility checking period	Sets the time in seconds after which UserGate sends OPTIONS request to the ICAP server to check if the server is available. Specify the value from 100 up to 600 seconds. **Example:** `100`
Skip if error occurs	If the option is enabled, UserGate does not send data to the ICAP server when the server is unavailable (does not respond to OPTIONS request). Enabling this option is not needed
Reqmod path	You can enable Reqmod mode by selecting the **Enable** flag.

Field	Description
	Then specify **path on ICAP server to work in Reqmod mode** in the format: `icap://icap-server:port/path` - specifying the full URI for the `reqmod` mode **Example**: `icap://FQDN:1344/reqmod`
Respmod path	Enabling the Respmod mode is not required
Respond with username	Set the **Enable** flag to enable sending user names to the ICAP server. Then specify **Headline** value that will be used for sending user names to the ICAP server. The default value is `X-Authenticated-User` If necessary, select the **Encode in base64** option to encode user names in base64. This can be needed if usernames contain characters of national alphabets
Respond with IP address	If the **Enable** flag is on, then sending user's IP address to the ICAP server is enabled. Then specify **Heading** that will be used for sending user's IP address to the ICAP server. The default value is `X-Client-Ip` Enabling this option is not needed
Respond with MAC address	If the **Enable** flag is on, then sending user's MAC address to the ICAP-server is enabled. Then specify **Heading** that will be used for sending user's MAC address to the ICAP-server. The default headline is `X-Client-Mac` Enabling this option is not needed

2. Create a balancing rule for ICAP servers (optional):
   a. Go to **Network policy** → **Load balancing**.
   b. Select **Add** → **ICAP Balancer** and fill in the fields:

Field	Description
Enable/Disable	Enables or disables the rule
Name	Rule name
Description	Rule description
ICAP servers	List of ICAP servers created in the previous step between which the load will be distributed

3. Create an ICAP rule:
    a. Go to **Security policies -> ICAP Rules**.
    b. Click **Add** and fill in the fields:

Field	Description
Enable/Disable	Enables or disables the rule
Name	Rule name
Description	Rule description
Operation	Possible options: • **Skip** - not send data to the ICAP server. By creating this rule, administrator can explicitly exclude specific kind of traffic from transmission to the ICAP server; • **Transmit** - send data to the ICAP server and wait for response. This is a standard operating mode for ICAP servers; • **Transmit and ignore** - send data to the ICAP server and ignore its response. In this case, no matter what the ICAP server responce would be, data sent to user will not be modified, but the ICAP server will receive a full copy of user's traffic.
ICAP servers	The list of ICAP servers created at the previous step between which the load will be distributed
Source	Traffic source zone and/or list of traffic source IP addresses
Users	The list of users and groups, the rule applies to. The following user types can be used: **Any**, **Unknown**, **Known**. To apply the rules to specific users or to the **Known** type, set up user identification
Destination	Traffic destination zone and/or lists of traffic destination IP addresses
MIME-type content	MIME-type lists. There is a possibility to manage video and audio content, images, executables, and other types. The administrator can also create custom groups of MIME-types
Categories	Lists of UserGate URL filtering categories
URL	URL lists
HTTP-method	A method used in HTTP requests. As a rule, POST or GET method is used
Service	Possible options: • **HTTP** - web traffic

Field	Description
	• **SMTP** - mail traffic. Emails will be sent to the ICAP server as a corresponding MIME type • **POP3** - mail traffic. Emails will be sent to the ICAP server as a corresponding MIME type

Warning!

ICAP rules are applied from top to bottom in the order they are listed in the console. Only the first rule for which the conditions defined in the rule are matched, is applied. This means that specific rules should be placed above general ones. Use **Up**/**Down** arrow buttons to change the order of applying rules.

Disabling ICAP

In case HTTP/HTTPS requests interception over ICAP is no longer required, you can disable the **iw_icap** module
To disable iw_icap module:

1. Stop Traffic Monitor processes:
   `iwtm stop`
2. Disable autostart for **iw_icap** process:
   `iwtm disable iw_icap`
3. Start Traffic Monitor processes:
   `iwtm start`
4. Check the state of Traffic Monitor processes:
   `iwtm status`

After you have disabled the module, restore original settings of the proxy server.

4.4.3 Configuring Licensing Server (in case of Cluster)

To receive correct data from interceptors in full, you need to configure the licensing server. It should run on the main Traffic Monitor server and be disabled on all other cluster nodes.

For this, take the following steps (on the node):

1. Stop **iw_licensed** service:
   `iwtm stop licensed`
2. Disable autostart of **iw_licensed** service:
   `iwtm disable iw_licensed`

And on the main Traffic Monitor server:

1. Enable autostart of **iw_licensed** service:
   `iwtm enable iw_licensed`
2. Run **iw_licensed** service:
   `iwtm start licensed`

To install, delete, request a license, or check its validity, please see the *InfoWatch Traffic Monitor. User Guide. Managing Licenses*.

4.5 Configuring Traffic Interception

The System can intercept data transferred via the following protocols:

Protocol	Description
SMTP (Simple Mail Transfer Protocol)	Mail protocol, which is used by mail clients for sending email messages to the server. It works in tandem with the POP3 protocol.
POP3 (Post Office Protocol, version 3)	Mail protocol, which is used by a mail client to receive incoming email messages from the server. For sending emails, the SMTP protocol is used.
HTTP (HyperText Transfer Protocol)	Data transfer protocol based on «client-server» technology, which is used to receive text information, stream video and sound from the website. You can realize interception of all transmitted data on the Internet (visited pages, portals, received and sent emails and messages).
HTTP(S)	HTTP protocol extension, which supports encryption. Data, transmitted over HTTP, are packed into cryptographic protocol, thereby providing data protection.
OSCAR	Instant and offline text messaging protocol, used by ICQ, AIM, Miranda, QIP programs. You can perform text messages interception, transmitted via these instant messaging programs. **Important!** ICQ traffic will be intercepted if a specific server that supports OSCAR protocol is used for ICQ.
NRPC	Mail Transfer Protocol in IBM Notes.
IMAP4 (Internet Mail Access Protocol)	Protocol of access to email. In accordance with POP3 protocol, it is used for work with emails but provides more broad work capabilities to work with a mailbox. For example, access to processing events, located on the server, as if they were located on local receiver computer, without the necessity of permanent file transfer with content from server and back. SMTP protocol is used to send messages. You can perform incoming emails interception, transmitted by email clients.

Depending on intercepted traffic type and the interception method, the System can be configured as follows:

- Configuring SMTP Traffic Interception;
- Configuring POP3 Traffic Interception;
- Configuring HTTP Traffic Interception;
- Configuring HTTPS Traffic Interception;
- Configuring ICQ Traffic Interception
- Configuring NRPC Traffic Interception;
- Configuring IMAP4 Traffic Interception;

- Receiving Objects Intercepted by InfoWatch Device Monitor.

Important!

It is assumed, that System is already installed before configuration starts.

For more information on installation for each deployment scheme, see the document *InfoWatch Traffic Monitor. Installation Guide*.

Important!

After you configure traffic interception, you should configure object analysis parameters (see document *InfoWatch Traffic Monitor. User Guide*).

Important!

On DM Agents installed on Astra Linux OS, the System only intercepts mail traffic sent via SMTP, POP3, and IMAP channels.

4.5.1 Configuring SMTP Traffic Interception

The section contains information on how to configure interception of SMTP traffic:

- SPAN, or Port Mirroring;
- Receiving Copies from Mail Server;
- Inline Mode;
- Configuring Mail Servers for SMTP Traffic Interception.

You can find a detailed description of SMTP traffic interception scheme in the *SMTP Traffic Interception Scheme* article.

SPAN, or Port Mirroring

1. Configure server (servers) for working with SPAN, or Port Mirroring (see *Configuring Server to Work with SPAN, or Port Mirroring Technology*).
2. Enable the autorun option for `iw_proxy_smtp` process which is responsible for intercepting SMTP traffic:
   `iwtm enable iw_proxy_smtp`
3. Make sure, that autostart of all processes, which participate in mail interception and analysis, is enabled:
   iw_sniffer, **iw_proxy_smtp**, **iw_messed**, **iw_warpd**, **iw_cas**, **iw_x2db** (see *Enabling and Disabling Process Autorun*).
4. To avoid duplication of intercepted data, disable **iw_smtpd** process (see *Enabling and Disabling Process Autorun*).

Receiving Copies from Mail Server

A rule for sending a shadow copy (BCC) of each sent message should be configured on the corporate mail server. The copy is sent to a nonexistent mail address with a nonexistent domain whose IP-address corresponds to Traffic Monitor Server. This functionality is supported by most mail servers.

Integration of Traffic Monitor Server with Postfix mail server is performed the following way. Postfix mail server, which is embedded in the Traffic Monitor system, receives copies of messages sent via SMTP. Copies are delivered to port **25** and transmitted to the **iw_smtpd** process on port **2025**. An event for each message is created in the System. Information about those events is then placed to the database.

To configure receiving copies from a mail server:

1. Make sure that **iw_smtpd** process autostart is enabled (see *Process autorun*).
2. On the mail server, configure forwarding of blind copies of SMTP messages to Traffic Monitor Server. Find a setup example for Microsoft Exchange Server 2007 and 2010 in *Forwarding Blind Carbon Copies with Microsoft Exchange Server 2007 and 2010*.
3. Configure the embedded Postfix server (see *Configuring Postfix Server in Traffic Monitor*).
4. Make sure that autostart of all processes which participate in traffic interception and analysis is enabled:
   postfix, **iw_smtpd**, **iw_messed**, **iw_warpd**, **iw_luaengined**, **iw_cas**, **iw_pas**, **iw_x2x**, **iw_x2db**, **iw_tech-tools** (see *Process autorun*).
5. Make sure that the value of the `"TransportModes"/" Sm tp"` parameter in the **/opt/iw/tm5/etc/luaengined.conf** is `Copy` or empty.

All emails received according to the rule of shadow copy or journal rule will contain a permanent recipient, for example, "`iwtm-mon@local`". To remove it, configure the `SmtpTmBccList` parameter in the **luaengined.conf** file.

> **Important!**
>
> If MS Exchange uses a journal rule, then, for correct processing of such emails, enable the setting in Traffic Monitor. For this, in the `messed.conf` configuration file, specify:
>
> `"ExchangeJournalReports": true,`
>
> To apply the changes, restart the service **iw_messed**.

Inline Mode

Traffic Monitor Server is used as an intermediate mail server. It includes Postfix mail server.

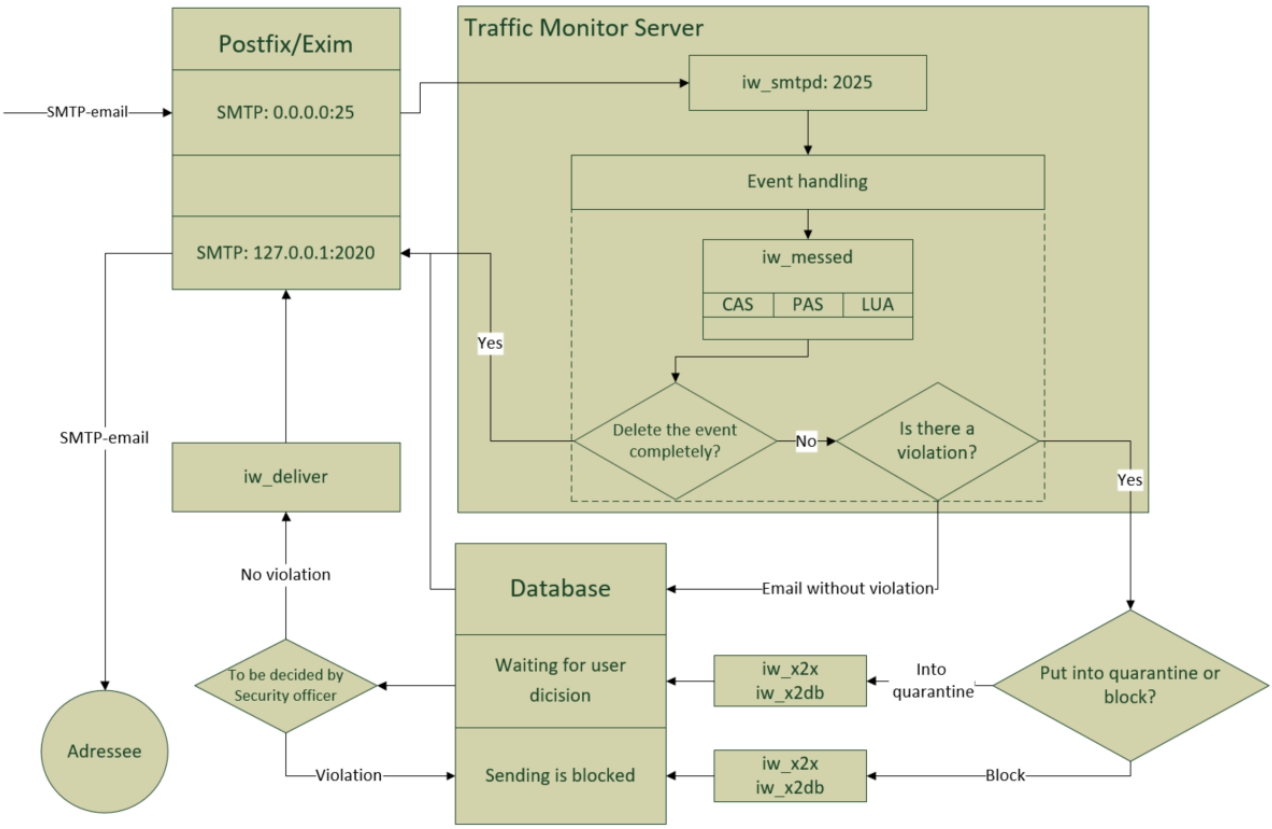
Integration of Traffic Monitor mail server with the mail server is implemented in the following way. Postfix mail server, which is embedded in the Traffic Monitor system, receives incoming SMTP messages via port **25**. SMTP messages are then transmitted to the **iw_smtpd** process via port **2025**. The System creates events for each email. Events are placed in the Traffic Monitor Database except when the *Delete event completely* action is selected in the transfer rule of the policy. In this case, the event is not placed in the Database and the email is sent to the recipient without analysis in Traffic Monitor.

In the **Inline** mode, SMTP traffic is analysed by Traffic Monitor and:

- If there is no violation, SMTP traffic is sent to the recipient or to the next relay server in the company's mail system by means of the embedded Postfix server. Delivery works as follows:

the **iw_messed** process sends emails to port **2020** of the Postfix mail server. The Postfix mail server either delivers emails to recipients directly or transmits them to the next mail relay server;

- If there is a violation and policies contain the *Quarantine* verdict, then the System sets the *Quarantine* verdict to the email. The email will be delivered only if the Security officer sets the *No violation* user decision and allows the delivery in the Management Console (see the document *InfoWatch Traffic Monitor. User Guide*). Delivery works as follows: the **iw_deliver** process transmits emails to port **2020** of the Postfix server that is embedded in the System. The Postfix mail server either delivers emails to recipients directly or transmits them to the next mail relay server;
- If there is a violation and policies contain the *Block* verdict, then the System sets the *Block* verdict to the email. In this case, the email will not be delivered.



To configure the inline mode for SMTP interceptor:

1. On the corporate mail server, configure Traffic Monitor as a relay server.

   > **Note:**
   >
   > If the company uses more than one mail server, each server should be configured to send messages to the embedded Postfix server.
   >
   > The configuration process may differ depending on the mail server which is used in the company.

   As a result, all mail traffic sent from a corporate mail server is transmitted to Traffic Monitor Server.
2. Make sure that autostart for the **iw_smtpd** and **iw_deliver** processes is enabled (see *Process autorun*).

3. Configure Postfix server embedded in the System (see *Configuring Postfix Server in Traffic Monitor*).
4. Stop the **iwtm** service:
   `iwtm stop`
5. Open the file **luaengined.conf**, located in `/opt/iw/tm5/etc`, for editing;
6. In `Smtp` property of `TransportModes` section, specify TM server name with interceptor (for example, `tm-server1`) and mode value `Normal` (*Block* mode):
   `"Smtp": [{"Server": "tm-server1", "Mode": "Normal"}]`

   > **Important!**
   >
   > In case of distributed installation, the mode is defined by the local **iw_luaengined** process or the one located on another node. That is why on each node in the `/opt/iw/tm5/etc/luaengined.conf` file in the `Smtp` property of the `TransportModes` section you need to specify several pairs of valid FQDN names of TM servers and required modes. For example:
   >
   > `"Smtp": [{"Server": "tm-server1.local", "Mode": "Copy"}, {"Server":"tm-server2.local.ru", "Mode": "Normal"}]`
   >
   > **Important!** The **iwssid.lua** and **luaengined.conf** files should be the same on all the nodes.

7. Save the file.
8. Start the **iwtm** services:
   `iwtm start`

For the traffic to be transmitted in spite of a verdict given:

1. In the code in step 6 above, set the `Transparent` value for the `TransportMode` parameter.
2. Save the file.
3. Restart the System:
   `iwtm restart`

Delivering SMTP emails with processing errors sent in "inline mode"

SMTP emails processing of which caused errors are to be put into the error queues: **queue/x2x-errors** and **queue/errors**. Delivering SMTP emails (including the ones with *Quarantine* verdict assigned) is performed by **iw_rammer** daemon from error queues. After that the queues are to be refined. The System extracts the sender and recipient's addresses (if provided) from an XML file. Otherwise, the email cannot be delivered. When sending an email to Security officer, the sender and recipient's addresses are specified within the configuration file **/opt/iw/tm5/etc/rammer.conf**. The email attachment contains XML and DAT object files. Email sending to the addressee as well as notification for Security officer is performed via Postfix mail server (port **2020**). Notification is to be delivered to Security officer via event email and the sender is specified from initial message. Connection data (address and port), as well as the addressee and sender addresses, are set within the **rammer.conf** configuration file (for details, see the document *InfoWatch Traffic Monitor. Configuration Files Guide. rammer.conf*).

SMTP email is to be delivered when meeting each of the following terms:

1. If it wasn't delivered in a regular mode (by **iw_messed** daemon);
2. If `"Block"` mode is assigned when processing the email and when the email got into error queue;
3. If the email wasn't qualified as a *Blocked* one.

Forced delivery measure is **/opt/iw/tm5/bin/rammer_tool** utility. To view the list of available settings, enter in the command line:

```
/opt/iw/tm5/bin/rammer_tool --help
```

Configuring Mail Servers for SMTP Traffic Interception

Chapter contains information on configuring mail servers to intercept SMTP traffic with Traffic Monitor:

- Forwarding Blind Carbon Copies with Microsoft Exchange Server 2007 and 2010;
- Configuring Blind Carbon Copies with ZCS Zimbra;
- Forwarding BCC with MDaemon;
- Configuring Postfix Server in Traffic Monitor.

Forwarding Blind Carbon Copies with Microsoft Exchange Server 2007 and 2010

To enable sending shadow copies of SMTP messages, a connector for sending messages to Traffic Monitor Server should be created and configured in the Exchange Management Console. For this, perform the steps below.

Creating SMTP connector that will send message copies to Traffic Monitor Server

1. Choose **Organization Configuration –> Hub Transport** node and press **New Send Connector**.
2. In the window of Connector creation wizard, type in connector name and choose the **Custom** type from the drop-down list. Click **Next**.
3. At the **Address space** step, create a new address space to transmit traffic. To do this, press  **Add**. In the dialog box that opens, specify address space parameters and press **OK**. Then click **Next**.
4. At the **Network Settings** step, choose **Route mail through the following smart hosts** and press  **Add**. In the opened dialog box, specify the IP address of the Traffic Monitor server, to which mail will be transmitted and press **OK**. Then click **Next**.
5. At the **Configure smart host authentication settings** step, choose **None**: by default Postfix does not require authorization on the Traffic Monitor Server. Click **Next**.
6. At the **Source Server** step, choose a server, the connector will be located on. Click **Next**.
7. At the **New Connector** step make sure, that connector parameters have been specified correctly, and click **New**. Wait for the creation process to be completed and click **Finish**.

The window of the Connector creation wizard will be closed.

Creating contact to which message copies will be sent

1. Choose **Recipient Configuration –> Mail Contact** node and press  **New Mail Contact**.
2. In the window of Contact creation wizard, choose **Mail Contact** and click **Next**.
3. At the **Configure smart host authentication settings** step, type in information about the contact.

   In the **External e-mail address** line, press  **Edit**. In the opened **SMTP Address** dialog box, type in mail address, which will receive message copies.

   > **Warning!**
   >
   > The mail contact domain should match the connector domain.

4. At the **New Mail Contact** step, make sure that connector parameters are specified correctly, and click **New**. Wait for process completion and click **Finish**.

The window of Contact creation wizard will be closed.

Warning!

For the System to function properly, it is necessary to have a Linux user with the same domain name as a new contact.

For example, if *user@company.com* contact is created, it is necessary that *user* should be among Linux clients.

To add a user on OS Linux, issue the following command under root:

```
useradd <username>
```

where <username> is a required user name. In the example above, issue the following command:

```
useradd user
```

Creating a transport rule for copying messages

1. Choose **Organization Configuration -> Hub Transport** node and press  **New Transport Rule**.
2. In the Transport rule creation wizard, type in a name of the rule being created and set a flag in the **Enable Rule** field. Click **Next**.
3. At the **Conditions**, select the **from users inside or outside the organization** checkbox. As a condition parameter, choose **Inside**. Click **Next**.
4. At the **Actions** step, select the **Blind carbon copy (Bcc) the message to address** checkbox. In the dialog box that opens, click the link with the **address** parameter and in the **Select Recipient** window, choose created contact. Click **OK**. In the wizard window, click **Next**.
5. You can configure exceptions at the **Exceptions** step if necessary. For example, for the messages from particular users not to be copied, select the **except when the message is from people** checkbox and choose desired users in the exception parameters. Click **Next**.
6. At the **Create Rule** step, make sure, that parameters have been specified correctly, and press **New**. Wait for the creation process to be completed and click **Finish**.

Blocking mail delivery when Traffic Monitor Server is not available

1. Choose **Organization Configuration -> Hub Transport** node and press  **New remote Domain**.
2. Specify the same name as the domain name used in item 3 step 2 of this chapter (creating contact, to which message copies will be sent).
3. Right-click on the new domain and choose **Properties** from the context menu.
4. In the window that opens, go to the **Message Format** tab.
5. Deselect the **Allow non-delivery reports** checkbox.
6. Click **OK**.

Important!

If MS Exchange uses a journal rule, then, for the correct processing of such emails, enable the setting in Traffic Monitor. For this, in the `messed.conf` configuration file, specify:

```
"ExchangeJournalReports": true,
```

To apply the changes, restart the service **iw_messed**.

Configuring Blind Carbon Copies with ZCS Zimbra

> **Warning!**
>
> All commands should be executed under **zimbra** user.

Configuring transport table

Use configuration block for Zimbra 5.0.9 or higher with the following string:

```
zmlocalconfig -e postfix_transport_maps="hash:/opt/zimbra/postfix/conf/
transportfile proxy:ldap:/opt/zimbra/conf/ldap-transport.cf"
```

> **Warning!**
>
> This command should be executed using **zmlocalconfig**. Editing **main.cf** will not be sufficient in the situation.

1. Check the existing transport table:
   `zmlocalconfig grep -i postfix_transport_maps`
   Get the command output of the following form:
   `postfix_transport_maps = proxy:ldap:/opt/zimbra/conf/ldap-transport.cf`
2. Create transport file:
   `mcedit /opt/zimbra/postfix/conf/infowatch`
3. Add the following line to file:
   `dlp.iw :[192.168.0.144]`
4. Check rights (**zimbra** should be owner of the group):
   `ll /opt/zimbra/postfix/conf/infowatch`
   `chown zimbra:zimbra /opt/zimbra/postfix/conf/infowatch`
5. Convert transport file to DB:
   `postmap /opt/zimbra/postfix/conf/infowatch`
6. Add transport file before the default file:
   `zmlocalconfig -e postfix_transport_maps="hash:/opt/zimbra/postfix/conf/infowatch proxy:ldap:/opt/zimbra/conf/ldap-transport.cf"`
7. Make sure, that relay_domains parameter of **main.cf** file contains all domains processed by server:
   `mcedit /opt/zimbra/postfix/conf/main.cf`
   Find or create line:
   `relay_domains = mydomain.com, mydomain.org, dlp.iw`

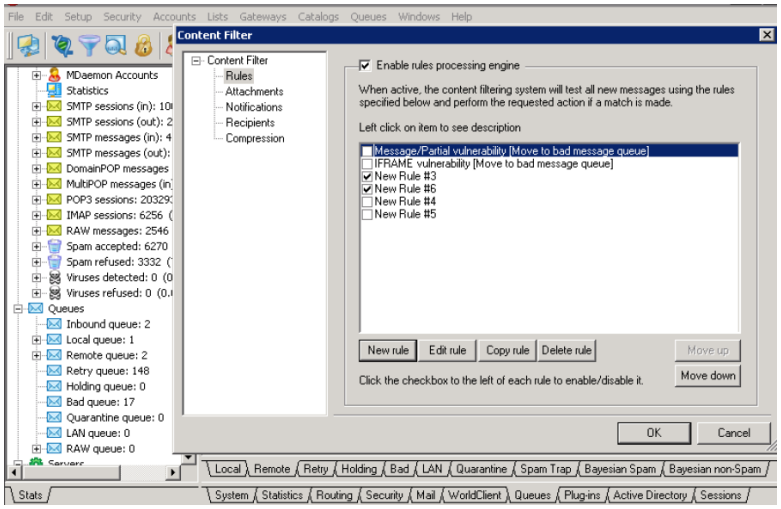
Configuring BCC

1. Add the following line to **/opt/zimbra/postfix/conf/main.cf** file:
   `/opt/zimbra/postfix/conf/main.cf`
2. Reboot Zimbra:
   `zmcontrol restart`

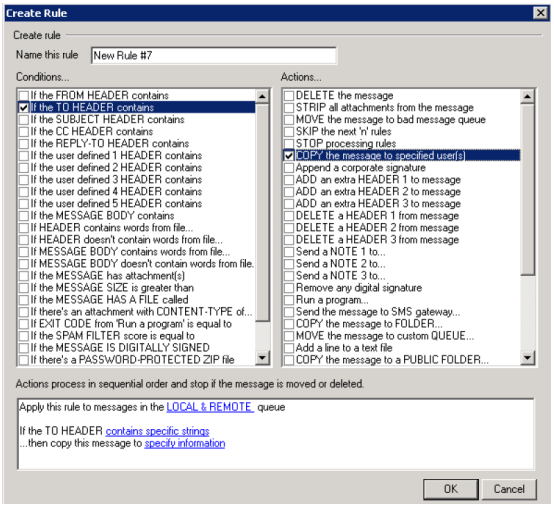
Forwarding BCC with MDaemon

In MDaemon mail client there is **Content Filter** option, which can be used to configure incoming/outgoing mail filtering.

1. Open MDaemon and go to **Security > Content Filter** tab.
2. Go to **Rules** tab. Press **New rule** button:



3. In the left column choose action condition. In the right column choose corresponding action:



4. Use blue sections to add values to rules.
5. Click **Add** and **OK**.

Configuring Postfix Server in Traffic Monitor

Depending on the System deployment method and on how the Traffic Monitor server is installed (on the same computer with the database or on different computers), you may need to configure the built-in Postfix server:

System deployment method		Configuring Postfix
SMTP traffic is delivered by the corporate mail server. Traffic Monitor receives only a copy of SMTP traffic. We recommend configuring Postfix for the traffic copy to be received properly	Traffic Monitor server is installed separately from the database	Required
	All components are installed on the same server	Configured by default
Traffic Monitor intercepts and delivers SMTP traffic by means of integration with Postfix	Traffic Monitor server is installed separately from the database	Required

System deployment method		Configuring Postfix
	All components are installed on the same server	Configured by default
The System receives a copy of SMTP traffic via Sniffer		Not required
Interception of SMTP traffic is not required (for example, Traffic Monitor is used only as Sniffer)		Not required

For proper interaction between Traffic Monitor and built-in Postfix mail server, make the following changes to `/etc/postfix/main.cf` and `/etc/postfix/master.cf` configuration files:

1. Enter the System on behalf of the `root` user:

   ```
   su - root
   ```

2. Apply these settings in the `main.cf` file:
   a. Set the value for `inet_interfaces` parameter as recommended on the Postfix site. In most cases, you should specify `all` as `inet_interfaces` parameter value.
   b. Set the value of the `message_size_limit` parameter (`5000000` by default) to be higher than what is set on the corporate mail server. Otherwise, senders may receive messages from Postfix that emails cannot be sent.
   c. Make sure that `mynetworks` parameter contains the correct name of the subnet where mail servers are located.
   d. In case Traffic Monitor participates in delivering SMTP traffic (see *Inline Mode*), configure the `relayhost` parameter. As its value, enter the IP address of the corporate mail server to which SMTP emails will be sent after analysis in Traffic Monitor and a password to access the mail server. For example:

      ```
      relayhost = [smtp.organization.ru]:port
      ```

3. Apply the settings in the `master.cf` file:
   a. Add the following line to enable sending mail to Traffic Monitor for analysis:

      ```
      smtp   inet  n    -    n   -   -   smtpd  -o content_filter=smtp:127.0.0.1:2025
      ```

      > **Note:**
      >
      > When the System is installed with the installer, this setting is applied automatically.

   b. In case Traffic Monitor participates in delivering SMTP traffic (see *Inline Mode*), write the following lines to the end of the file:

      ```
      127.0.0.1:2020   inet  n   -   n  -   21   smtpd
           -o content_filter=
      ```

```
-o local_recipient_maps=
-o relay_recipient_maps=
-o smtpd_restriction_classes=
-o smtpd_client_restrictions=
-o smtpd_helo_restrictions=
-o smtpd_sender_restrictions=
-o mynetworks=127.0.0.0/8
-o strict_rfc821_envelopes=yes
-o smtpd_error_sleep_time=0
-o smtpd_soft_error_limit=1001
-o smtpd_hard_error_limit=1000
-o myhostname=<fully_qualified_domain_name_of_the_server>
```

Important!

Lines that specify the service options should begin with a space. For example:

" `-o <option_specification>` "

4. Check that the syntax in configuration files is correct:

```
postfix check
```

5. Restart Postfix:

```
systemctl restart postfix
```

6. Check the availability of the Postfix mail server via port 25 with the following command:

```
telnet <TM_IP> 25
```

where `<TM_IP>` is the IP address of the Traffic Monitor server on which Postfix has been configured.
In response to this command, the following message should be displayed:

```
Connected to mail.server.com
```

Note:

If Telnet client is not installed, you can install it as MS Windows 2008–2012 component or use the PuTTY application instead.

If you cannot establish the connection, perform step-by-step checks of the subnet and port settings depending on your network configuration.

Important!

Traffic Monitor installed on a server running Alt Server 10 does not support interception of emails with UTF-8 encoded headers. To set it up, reinstall the Postfix server with the `libicu-devel` package. To learn more, see the Postfix site.

4.5.2 Configuring POP3 Traffic Interception

1. Configure server (servers) for working with SPAN, or Port Mirroring (see *Configuring Server to Work with SPAN, or Port Mirroring Technology*).
2. Make sure, that `Hi Port` parameter of `capstack` section in **/opt/iw/tm5/etc/sniffer.conf** configuration file has value `110`:

```
"ListenAreas": {
"capstack": {
"Balancer": "IpPref",
"Interface": "eth1",
"MaxClient": 16,
"QueueMemorySizeInBytes": 67108864,
"Rules": [
{
"HiPort": 110,
"IP": "0.0.0.0",
"LoPort": 110,
"Mask": "0.0.0.0",
"Policy": "ACCEPT"
```

3. Specify value `true` for `pop3` parameter of **/opt/iw/tm5/etc/capstack.conf** configuration file.

4.5.3 Configuring HTTP Traffic Interception

For this traffic interception method, see *HTTP Traffic Interception Scheme*.

SPAN, or Port Mirroring

1. Configure server (servers) to work with SPAN technology, or Port Mirroring (see *Configuring Server to Work with SPAN, or Port Mirroring Technology*).
2. Enable autostart of HTTP traffic interceptor:
   `iwtm enable iw_proxy_http`
3. Make sure, that autostart of all processes, which participate in traffic interception and analysis, is enabled:
   iw_sniffer, iw_warpd, iw_cas, iw_x2db (see *Enabling and Disabling Process Autorun*).

Inline Mode

Note:

In this item, «inline» mode of interception is described, which is being transmitted by ICAP protocol – do not confuse with «inline» mode for SMTP traffic.

To configure inline transfer mode, issue the following actions:

1. Learn this working schema (see *Intercepting Traffic, Transmitted via ICAP Protocol*).
2. Configure HTTP requests interception parameters on proxy server (see *Configuring ICAP*).

4.5.4 Configuring HTTPS Traffic Interception

For this traffic interception method, see in *HTTPS Traffic Interception Scheme*.

Inline Mode

Note:

In this item, «inline» mode of interception is described, which is being transmitted by ICAP protocol – do not confuse with «inline» mode for SMTP traffic.

To configure inline transfer mode, issue the following actions:

1. Learn this working schema (see *Intercepting Traffic, Transmitted via ICAP Protocol*).
2. Configure HTTP requests interception parameters on proxy server (see *Configuring ICAP*).
3. Enable HTTPS traffic parsing on proxy-server.

   Note:

   This setting depends on proxy-server modification. You can get actual information on configuring proxy-server from its manual.

4. Add certificate to the list of trusted root browser certificates of each Workstation, where outgoing HTTPS traffic is supposed to be controlled.

   Note:

   Get configuration information from browser documentation.

4.5.5 Configuring ICQ Traffic Interception

For this traffic interception method see *ICQ Traffic Interception Scheme*.

SPAN, or Port Mirroring

1. Configure server (servers) to work with SPAN technology, or Port Mirroring (see *Configuring Server to Work with SPAN, or Port Mirroring Technology*).
2. Enable autorun of ICQ traffic interceptor:

```
iwtm enable iw_proxy_icq
```

4.5.6 Configuring NRPC Traffic Interception

This type of traffic interception is described in the article *NRPC Traffic Interception Scheme*.

SPAN, or Port Mirroring

1. Configure server (servers) to work with SPAN technology, or Port Mirroring (see *Configuring Server to Work with SPAN, or Port Mirroring Technology*).
2. Open **capstack.conf** configuration file, which is located in `/ort/iw/tm5/etc` directory.
3. Make sure, that `Usedprocessors` section has NRPC traffic interceptor enabled:
   ```
   "UsedProcessors": {
   "nrpc": true,
   ```
4. Make sure, that `Processors` section has NRPC traffic transmission enabled:
   ```
   "processors": {
   "nrpc": {
   "TrackExternal": true,
   "TrackInternal": true
    },
   ```
5. Save changes in the **capstack.conf** file.
6. Enable autostart for **iw_capstack** process:
   `iwtm enable iw_capstack`
7. Restart **iw_capstack** process:
   `iwtm restart capstack`

4.5.7 Configuring IMAP4 Traffic Interception

1. Configure server (servers) for working with SPAN, or Port Mirroring (see *Configuring Server to Work with SPAN, or Port Mirroring Technology*).
2. Make sure, that `Hi Port` parameter of `capstack` section in **/opt/iw/tm5/etc/sniffer.conf** configuration file has value `143`:
   ```
   "ListenAreas": {
   "capstack": {
   "Balancer": "IpPref",
   "Interface": "eth1",
   "MaxClient": 16,
   "QueueMemorySizeInBytes": 67108864,
   "Rules": [
   {
   "HiPort": 143,
   "IP": "0.0.0.0",
   "LoPort": 110,
   "Mask": "0.0.0.0",
   "Policy": "ACCEPT"
   ```
3. Specify value `true` for `imap4` parameter of **/opt/iw/tm5/etc/capstack.conf** configuration file.

4.5.8 Receiving Objects Intercepted by InfoWatch Device Monitor

1. Install and configure Traffic Monitor Server part (see document *InfoWatch Traffic Monitor. Installation Guide. Installing Traffic Monitor Server and Database*).
2. Setup event transmission from Device Monitor to Traffic Monitor server (see document *InfoWatch Device Monitor. User Guide. Device Monitor Server Settings. Connecting to Traffic Monitor Server*).
3. Setup object analysis parameters (see document *InfoWatch Traffic Monitor. User Guide. Technologies Section*).

4.6 Process autorun

This section contains features of System's processes autostart:

- Check of Process Autorun;
- Enabling and Disabling Process Autorun.

General information on working with processes is available in chapter *Traffic Monitor Server Processes*.

4.6.1 Check of Process Autorun

Important!

Disable autorun of unlicensed processes responsible for traffic interception. This allows you to reduce the number of messages in the log.

Note:

For instructions on enabling and disabling process autorun, see *Enabling and Disabling Process Autorun*.

When you divide servers into the main server, interception server and web console server, make sure that autorun is configured according to notes in the table.

autorun must be enabled only for processes, which this copy of Traffic Monitor server needs for proper operation:

Process	By default autorun is enabled
`iw_adlibitum`	Yes **Note:** The process should operate only on the main server
`iw_agent`	Yes
`iw_analysis`	Yes
`iw_kicker`	Yes **Note:** The process should operate only on a server with the web console

Process	By default autorun is enabled
`iw_blackboard`	Yes
`iw_bookworm`	Yes **Note:** The process should operate only on the main server
`iw_capstack`	No
`iw_cas`	Yes
`iw_configerator`	Yes **Note:** The process should operate only on a server with the web console
`iw_deliver`	Yes **Note:** The process should operate only on the main server
`iw_icap`	Yes
`iw_is`	Yes **Note:** The process should operate only on the main server
`iw_indexer`	Yes **Note:** The process should operate only on the main server
`iw_image_autoling`	Yes **Note:** The process should operate only on a server with the web console
`iw_text_autoling`	Yes **Note:** The process should operate only on a server with the web console
`iw_licensed`	Yes **Note:** The process should operate only on the main server
`iw_luaengined`	Yes
`iw_messed`	Yes
`iw_pas`	Yes

`iw_proxy_http`
`iw_proxy_icq`
`iw_proxy_smtp`
	No

`iw_qmover_client`	No
`iw_qmover_server`	No

Process	By default autorun is enabled
`iw_sample_compiler`	Yes
`consul`	Yes
`iw_smtpd`	Yes
`iw_sniffer`	No
`iw_system_check`	Yes
`iw_tech_tools`	Yes
`iw_updater`	Yes
`iw_warpd`	Yes
`iw_x2x`	Yes
`iw_x2db`	Yes
`iw_xapi_xapi` `iw_xapi_puppy`	Yes
`iw_space_manager`	Yes **Note:** The process should operate only on the main server
`iw_attributor`	No
`iw_rest2push`	No **Note:** You have to start the process manually. For details, see *Enabling and Disabling Process Autorun*

In order to check which processes are configured to start automatically, use the command:

```
iwtm status
```

This will display a list of all Traffic Monitor process with their statuses. In `enabled state` field, find autorun status:

- `loaded (enabled)` if autorun is enabled;
- `loaded (disabled)` if autorun is disabled;
- `loaded (static)` if service is in static mode, i.e. it runs according to its settings;
- `masked (masked)` or `masked (bad)` if service is masked, i.e. autorun is disabled.

In order to check autorun status for a specific service, use the command:

```
iwtm status <process_name>
```

Note:

To delete all `iw_kicker` processes from the server, delete `iwtm-web-meta` package:

```
rpm -e iwtm-web-meta
```

4.6.2 Enabling and Disabling Process Autorun

In order to enable/disable process autorun, use the commands:

- **Enabling process autorun**

  ```
  iwtm enable <process_name>
  ```

- **Disabling process autorun**

  ```
  iwtm disable <process_name>
  ```

- **Enabling autorun for several processes**

  ```
  iwtm enable <process1_name> <process2_name> ... <processN_name>
  ```

- **Disabling autorun for several processes**

  ```
  iwtm disable <process1_name> <process2_name> ... <processN_name>
  ```

- **Enabling autorun for all processes**

  ```
  iwtm enable all
  ```

- **Disabling autorun for all processes**

  ```
  iwtm disable all
  ```

Example of enabling/disabling `iw_proxy` service:

The `iw_proxy` service includes three components: `iw_proxy_http`, `iw_proxy_icq` and `iw_proxy_smtp`. For this reason:

1. For each process of the service, enable/disable autorun:

- **Enabling autorun**

```
iwtm enable iw_proxy_http iw_proxy_icq iw_proxy_smtp
```

- **Disabling autorun**

```
iwtm disable iw_proxy_http iw_proxy_icq iw_proxy_smtp
```

2. Restart the processes of `iw_proxy` service:

```
iwtm restart iw_proxy_http iw_proxy_icq iw_proxy_smtp
```

All Traffic Monitor processes with the `loaded ( enabled)` status can run automatically, while processes with the `masked (masked )` status can only be started manually.

In order to configure the process autorun:

1. Go to the directory with unit files, `/usr/lib/systemd/system`.
2. Edit the process unit file and save your changes. For details, see *InfoWatch Traffic Monitor. Configuration Files Guide*. *Traffic Monitor configuration and unit files description*.
3. Reload the configuration:

```
systemctl daemon-reload
```

4. Add the configured process to autorun:

```
iwtm enable <process_name>
```

5. Start the process:

```
iwtm start <process_name>
```

6. Check if the process is started with one of the following commands:

- **Command to check status of a specific process**

```
iwtm status <process_name>
```

- **Command to check status of all services**

```
iwtm status
```

In `Active` field or in the line corresponding to the process, find the process status:

- `active (running)`;
- `inactive (dead)`.

4.7 Remote database interaction module

In case the System deployment scheme is chosen in a way, that database, receiving intercepted objects is located on a remote server, you need to install a remote database interaction module.

The remote database is a database installed on a remote server with **TME DB Server** or **TME All-in-one** installation key (for details on installation keys, see *InfoWatch Traffic Monitor. Installation Guide. Distributed Installation of TM Enterprise*).

The standard process of data transmission to a database is **iw_x2db** (see *List of Traffic Monitor Server End Processes*). Using a remote database integration module with a remote database provides an additional opportunity to configure:

- the speed of files transmission to file queue;
- schedule of the width of the file transmission channel.

Configuration remote database integration module includes the following steps:

- Configuring the Client of the Remote Database Management Module;
- Configuring the Server of the Remote Database Management Module.

4.7.1 Configuring Data Collection in a Branch Network

If the System is deployed in an extensive branch network, then the following services are responsible for sending data on intercepted events and receiving this data on Traffic Monitor server:

- **iw_qmover_client** - sends data from branches;
- **iw_qmover_server** - receives data in the head office.

By default, both services are disabled and can be enabled manually by the Security Officer from Traffic Monitor Management Console:

- **iw_qmover_server** service - the first one;
- **iw_qmover_client** service - the second one.

These services can be configured in the directory `/opt/iw/tm5/etc:` **qmover_client.conf** and **qmover_server.conf** configuration files (for details, see the documents *InfoWatch Traffic Monitor. Configuration Files Guide. qmover_client.conf and iw_qmover_client.service* and *InfoWatch Traffic Monitor. Configuration Files Guide. qmover_server.conf and iw_qmover_server.service*).

4.7.2 Configuring the Client of the Remote Database Management Module

Configuring general settings

1. Configure the following client settings in **qmover_client.conf** configuration file, located in `/opt/iw/tm5/etc` directory:

Parameter	Description
`ListenerIP`	Server IP address (**qmover_server** service)
`Port`	Server port (central office)
`NookDir`	**qmover_server** service working catalogue

`ChannelWidth`	Channel bandwidth, speed of data downloading to Traffic Monitor (Kbps). Can be implicitly limited with `WindowSize` parameter

2. Restart Traffic Monitor:
   `iwtm restart`

Configuring processes autorun

- Enable autorun for **iw_qmover_client** process.
- Disable autorun for **iw_x2db**, **iw_x2x**, **iw_deliverd** and **iw_adlibitum** processes.

For more information on disabling and enabling processes autorun see *Enabling and Disabling Process Autorun*.

Changing the bandwidth of the data transmission channel

By default, the bandwidth is 256 Kbps. You can configure the bandwidth to be changed automatically at different time periods. To do this, use **iw_qmover_channel_width_setter** utility.
Bandwidth change is allowed. The bandwidth can be varied from 10 Kbps through 2 Mbps.

To set bandwidth limitations,
Run the utility with the mandatory parameters:
`iw_qmover_channel_width_setter <unix_socket_name> <bandwidth>`
where

- `unix_socket_name` socket name (is being created automatically on service start; is being automatically deleted on service stop); default value - `/opt/iw/tm5/run/.channel_socket` ;
- `bandwidth` the bandwidth (Kbps) to be set.

Example
There is a channel whose bandwidth is 256 Kbps. It is required that only 50% of the channel can be used from 9 AM to 6 PM. And from 6 PM to 9 AM – 100%.
Calculate the bandwidth for the required bandwidth usage. In this example, the bandwidth for different time intervals is following:

Interval	**Percent of the channel width**	
	Per cent of the channel width	Equivalent in Kbps
9 AM to 6 PM	50%	128
6 PM to 9AM	100%	256

Add to **etc/crontab** these commands:

```
00 9 * * * iwtm /opt/iw/tm5/bin/iw_qmover_channel_width_setter /opt/iw/tm5/run/.channel socket 128
 00 18 * * * iwtm /opt/iw/tm5/bin/iw_qmover_channel_width_setter /opt/iw/tm5/run/.channel_socket 256
```

4.7.3 Configuring the Server of the Remote Database Management Module

Configuration setting

Configure the following client settings in the configuration file **/opt/iw/tm5/etc/qmover_server.conf**. The directory where the objects queue is stored is /opt/iw/tm5/queue/db. For this:

1. Specify the IP address of a client (branch office) in the `ip` parameter.

2. Specify the port, where the server listens for objects coming from clients in the `port` parameter.

> **Warning!**
>
> When branch offices parameters are changed (offices quantity or IP addresses), manually edit **qmover_server.conf** file.

Configuring autorun

Enable autorun for **iw_qmover_server** process. For more information on enabling and disabling processes autorun see *Enabling and Disabling Process Autorun*.

4.8 Configuring OCR Extractors

OCR for different interceptors is configured in the configuration files:

- **To enable OCR for analysis of intercepted images**, use the `/opt/iw/tm5/etc/warpd.conf` file. In the `Warp` section, set the value of the `EnableOCR` parameter to `true`.
- **To enable OCR for analysis of images loaded as sample documents**, use the `/opt/iw/tm5/etc/sample_compiler.conf` file. Set the value of the `EnableOCR` parameter to `true`.

To configure OCR for different interception channels, use the `/opt/iw/tm5/etc/config-perm/bookworm/ocr_custom.xml` file. Setting levels are listed by priority, from lowest to highest.

- Level of a service;
- Level of the event type for a particular service;
- Level of a protocol for a particular event type.

For details, see *Configuring OCR*.

After you have changed the files, run the following command:

```
iwtm restart
```

> **Note:**
>
> For distributed installation when you install Traffic Monitor on several servers, you should specify individual OCR settings for each server.

> **Note:**
>
> If a new event type is registered by means of SDK, it will be processed with settings specified for the service the event type belongs to. If a new protocol is registered, it is processed with settings specified for the event type this protocol belongs to.

4.8.1 Interception channels

The table below lists interception channels for which OCR can apply.

Service	Event type	Protocol
Email	Email	POP3 SMTP IMAP MAPI NRPC
	Web email	HTTP HTTPS
Messenger	Skype	SKYPE
	ICQ	OSCAR
	XMPP	XMPP
	MS Lync	SIP
	Telegram	-
	VK (VKontakte)	HTTPS
	Facebook	HTTPS
	WhatsApp	WhatsApp
Internet activity	Web message	HTTP HTTPS
File exchange	FTP	FTP
	External devices	-
	Cloud storages	HTTPS
Printer and MFD	Print	-
Storage	Data Discovery	-
Recording media	Photo	-

4.8.2 Supported OCR extractors

During Traffic Monitor installation, both extractors are installed to the server end of the System, and one of them is chosen for usage:

- ABBYY FineReader Engine 12;
- Tesseract 4.1.1.

In case you selected ABBYY FineReader Engine, configure the license manually.

Functional differences of extractors are listed in the table below:

Functionality	ABBYY FineReader Engine	Tesseract
Recognition of image rotation angles	0 (+/-20) , 90 (+/-20), 180 (+/-20) and 270 (+/-20) degrees	-
Recognition of colour images	+	-
Image correction	+	-
Recommended image resolution	300 DPI for text with a font size from 10pt 400-600 DPI for text with a font size of 9pt and less	300 DPI for text with a font size from 10pt 400-600 DPI for text with a font size of 9pt and less

Note:

To avoid deviation from recommended image resolution, when using ABBYY FineReader Engine the parameter `AutoOverwriteResolution` (`true` by default) is used, which allows to specify the image resolution automatically.

Tesseract settings

Select the extractor

To configure the System to use Tesseract:

1. Delete the `/opt/iw/tm5/bin/iw_image2text` symlink.
2. Create a symlink:

```
ln -s /opt/iw/tm5/bin/iw_image2text_ts /opt/iw/tm5/bin/iw_image2text
```

Change image size limits

To change size limit for images that can be sent:

1. Go to `/ opt/iw/tm5/etc/image2text_ts.conf` directory.
2. Edit the `MaxSizeInKb` (maximum size) and `MinSizeInKb` (minimum size) parameters. By default, their values are 1536 KB and 40 KB respectively.

Enable OCR only for cloud storage events

To enable OCR only for cloud storage events:

1. Open the `/opt/iw/tm5/etc/config/bookworm/services.xml` file and find `mnemo` and `key` parameters, which are suitable for cloud storage.
   For example:

   ```
   object_type mnemo="cloud_storage"

   key="AA9DFB259F0DFEE040BADC95815E13A200000000"
   ```

2. Copy the values of the `key` and `mnemo` parameters and paste them into `/opt/iw/tm5/etc/config-perm/bookworm/ocr_custom.xml` file.

ABBYY FineReader Engine settings

Configure license

To configure ABBYY FineReader Engine license:

1. In the `/opt/iw/tm5/etc/image2text_fre.conf` configuration file, edit values of the fields as follows:
   - Enter serial number in the `SerNum` field;
   - Enter password in the `Pwd` field.
2. Copy the license file (`.LocalLicense` format) to the `/var/lib/Content_AI/SDK/12/Licenses` directory.
3. Make sure that `iwtm` user can access the downloaded file.
4. Restart the `iw_warpd` service.

Select the extractor

To configure the System to use ABBYY FineReader Engine:

1. Delete the `/opt/iw/tm5/bin/iw_image2text` symlink.
2. Create a symlink:

   ```
   ln -s /opt/iw/tm5/bin/iw_image2text_fre /opt/iw/tm5/bin/iw_image2text
   ```

Activate fast mode of document processing

ABBYY FineReader Engine can operate in two modes:

- *Balanced mode* provides the best quality of images processing (default mode). You can configure the user profile of the mode in `/opt/iw/tm5/etc/CRProfile.ini` file;
- *Fast mode* provides the fastest images processing, 64 pages can be loaded at a time while document synthesis. You can configure the user profile of the mode in `/opt/iw/tm5/etc/CRProfile_fast.ini` file. If required, configure the mode manually.

The following options are enabled while *Fast mode* is on:

- Fast mode of page orientation detection (the quality lowers);
- Standard analysis of page, no sample selection;
- Ignoring the shadows;
- Detecting tables during analysis;
- Fast objects extraction (the quality lowers).

The following options are disabled while *Fast mode* is on:

- Balanced mode;
- Processing images as photos;
- Compressing images;
- Using image binarization;
- Detecting font parameters;
- Removing garbage from image;
- Removing texture from temporary image;
- Detecting porous text;
- Detecting whole text on image, including text within image;
- Extracting whole text from image;
- Detecting headlines;
- Detecting footnotes;
- Detecting table of contents;
- Detecting running titles;
- Detecting pages of document.

To activate the fast mode of document processing, in `"ABBYYProfile":"etc/CRProfile_fast.ini"` parameter of the configuration file `/opt/iw/tm5/etc/image2text_fre.conf`, indicate the path to the file with user profile configuration, then save the changes.

> **Warning!**
>
> Speed optimization of images processing cannot be applied for scanned copies of schemes, location maps, monochrome images.

Change image size limits

To change size limit for images that can be sent:

1. Go to `/opt/iw/tm5/etc/image2text_fre.conf` directory.
2. Edit the `MaxSizeInKb` (maximum size) and `MinSizeInKb` (minimum size) parameters. By default, their values are 1536 KB and 40 KB respectively.

Enable OCR only for cloud storage events

To enable OCR only for cloud storage events:

1. Open the `/opt/iw/tm5/etc/config/bookworm/services.xml` file and find `mnemo` and `key` parameters, which are suitable for cloud storage.
   For example:

   ```
   object_type mnemo="cloud_storage"

   key="AA9DFB259F0DFEE040BADC95815E13A200000000"
   ```

2. Copy the values of the `key` and `mnemo` parameters and paste them into `/opt/iw/tm5/etc/config-perm/bookworm/ocr_custom.xml` file.

4.9 Configuring Notifications for Users and Employees

Users of the Management Console can configure sending notifications from the System to users or employees. To enable notifications:

- specify the email address from which notifications will be sent to users or employees;
- configure sending emails via Postfix mail server

Notifications are sent when policy rules are triggered (for details, see the document *InfoWatch Traffic Monitor. User Guide*).

To check if emails are sent via the mail server:

Run the following command:

```
sendmail -v <employee@company.com> < <sample.log>
```

where:

- `<employee@company.com>` - email address of the notification recipient;
- `<sample.log>` - a text file that will be used as the email text (we recommend using a simple text file up to 1 MB).

If emails are not sent, configure the delivery via Postfix.

To configure the mail delivery via Postfix:

1. Open the **/etc/postfix/main.cf** file.
2. Specify the required mail server as the `relayhost` value:
   `relayhost = <mail.company.ru>`
   `where <mail.company.ru> - the required mail server`

   > **Note:**
   >
   > If the **/etc/postfix/main.cf** file does not have a line containing the `relayhost` attribute, add this line.

3. Reboot Postfix using the following command:
   `service postfix restart`
4. Check the mail delivery using the following command:
   `sendmail -v <employee@company.com> < <sample.log>`
   where:

- `<employee@company.com>` - email address of the notification recipient;

- `<sample.log>` - a text file that will be used as the text of the email (we recommend using a simple text file up to 1 MB).

4.10 Limiting the number of events that can be found

You can specify a limitation for the maximum number of events that will be displayed in the Management Console as a result of applying filter. For this, in the *SETTING* table of a database, specify a desired value for the *query_stop_count* attribute. The default value is 10.000.

4.11 Service-to-service cooperation setup (Consul service)

To register the services, monitor the accessibility and detect Traffic Monitor components the decentralized fault-tolerant discovery service Consul is used. The Consul agent:

- is installed on each host and is a significant part of cluster activity;
- revealing the services, collecting the services state data, realizing DNS, API HTTP and RPC CLI interfaces;
- can be started in one of two modes: client and server one.

Consul is used in all types of installation. Information on System installation is available in *InfoWatch Traffic Monitor. Installation Guide. System Deployment Schemes and Installation Types*.

When using All-in-One installation mode, the configuration parameters to connect to the Consul service are set automatically during the installation. The only exception is the distributed installation, when Traffic Monitor and the database are installed on different servers (when the System is installed on more than one node and has more than one network interface). See how to configure in *Configuring Consul and creating a cluster* article.

The service operation settings are to be done within the configuration file located by the following path: `/opt/iw/tm5/etc/consul/consul.json`. The full description of parameters is available at the page *consul.json configuration file and iwtm-consul.service unit file*.

4.11.1 Starting and stopping the service

To start the Consul service, use one of the following commands:

- ```
  service iwtm-consul start
  ```
- ```
  systemctl start iwtm-consul.service
  ```

To stop the Consul service, you can use the following commands:

- ```
  service iwtm-consul stop
  ```
- ```
  systemctl stop iwtm-consul.service
  ```

If necessary, you can start the Consul agent **manually**:

```
consul agent -data-dir=<path> -bind=<bind_addr> -bootstrap -server -ui
```

where:

- `<path>` is a path to the directory with service data (e.g., `/opt/iw/tm5/var/consul`);
- `<bind>` is the network interface address.

If the System has more than one network interface, specify one of the following parameters:

- `-bind=<ip-address>`;
- `-config-file` or `-config-dir` – the path to the configuration file (e.g.: `/opt/iw/tm5/etc/consul`).

If several configuration files are loaded, their options are merged.

4.11.2 Registering services in Consul

You can register a service in Consul in 2 ways:

- using HTTP API or agent's configuration file, in the case, when service can communicate to Consul independently;
- registering service as an external component.

Services with obligatory registry in Consul	Services installed on nodes with Consul Server or Consul Client
iw_adlibitum	iw_adlibitum
iw_agent	iw_agent
Iw_analysis	Iw_analysis
iw_blackboard	iw_bookworm
iw_bookworm	iw_capstack
iw_capstack	iw_cas
iw_cas	iw_configerator
iw_icap	iw_icap
iw_deliver	iw_indexer
iw_indexer	iw_licensed
iw_licensed	iw_kicker
iw_luaengined	iw_luaengined
iw_messed	iw_messed
iw_pas	iw_pas
iw_proxy_http	iw_proxy_http
iw_proxy_icq	iw_proxy_icq
iw_proxy_smtp	iw_proxy_smtp
iw_qmover_server	iw_qmover_server
iw_sample_compiler	iw_sample_compiler

Services with obligatory registry in Consul	Services installed on nodes with Consul Server or Consul Client
iw_system_check	iw_system_check
iw_smtpd	iw_smpd
iw_tech_tools	iw_tech_tools
iw_updater	iw_updater
iw_warpd	iw_warpd
iw_xapi_xapi	iw_xapi_xapi
iw_xapi_puppy	iw_xapi_puppy
iw_x2x	iw_x2x
iw_x2db	iw_x2db
	Web (backend)

After the registration, Traffic Monitor services and Consul interact the following way:

- registration using Consul client;
- establishing TCP connection or returning HTTP code;
- deregistration.

4.11.3 Distributed installation

Important!

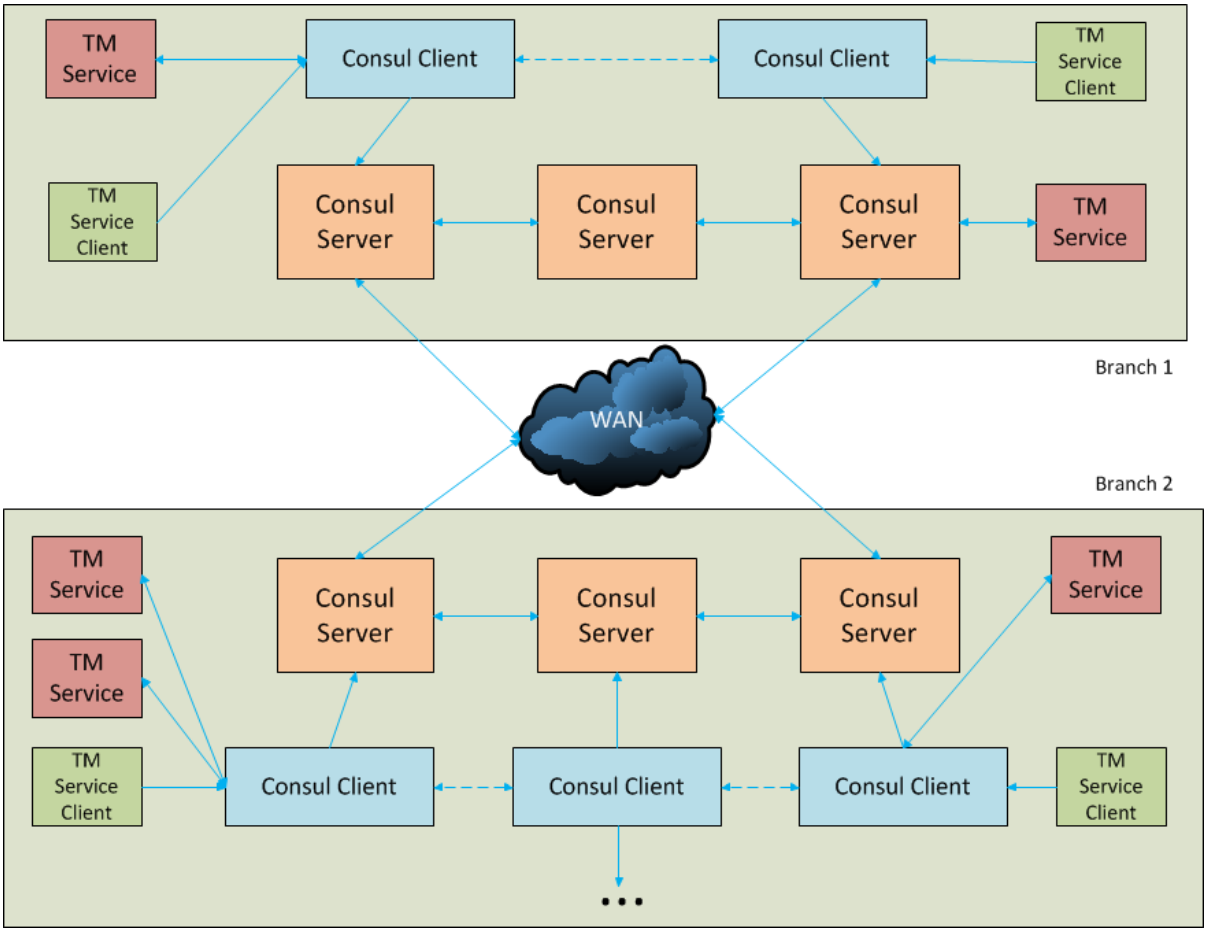
To use the Consul service, the full-mesh network topology ("each-to-each" connection) should be implemented for all the agents within the cluster and for servers, when combining them into WAN.

Clustering

The client collects the node- and service-related data and sends it to the server. System components searching for services send the request to the agent started on a local workstation, which resends the request to an available server. If the server is unable to respond to the request, it can send the request to a different data center (a branch) and retrieve the response.

Servers compose a cluster and define independently the leading server, responsible for elements coordination within the cluster. The first/the only one server is usually started in bootstrap mode (assigning the leader manually). When starting the agent, to connect to cluster it is enough specifying one server of this cluster. While configuring it in `retry_join` option, it is recommended specifying all the servers of the cluster. A quorum to hold the operations and to provide the consistency is required in each data center. If there are few data centers, a separate cluster to be created in each of them. Data centers are not isolated from each other due to service detection task. An agent within one data center can get information from a separate data center.

The Consul network can use one server, but to avoid data loss it is recommended using from three to seven servers within the data center.



Example:

A cluster consisting of three nodes (servers) continues operating when one server is out of operation.

Below are the examples of using the interface command line:

Command	Description
`consul members`	Displays the list of cluster members
`consul catalog services -node <node_name>`	Displays the list of cluster servers on a specific computer
`consul operator raft list-peers -stale=true`	Displays the list of all the servers regardless of the cluster

Connecting to cluster

You may register extra nodes on the cluster using one of the following ways:

- Enter the command:

```
consul join <ip-address>
```

Please make sure that, within `consul.json` configuration file, all the servers and Consul clients have the same parameters `datacenter` and `encrypt` (see *consul.json configuration file and iwtm-consul.service unit file*)

- In `consul.json` configuration file, in `retry_join` parameter specify the list of servers to be joined

4.11.4 Network rules configuration to access Consul

To set up the network exchange the following ports between all Consul agents (servers and clients) should be accessible.

Port by default	Interface	Supported protocols	Description
8300	Server RPC	TCP	Used by servers to process incoming queries from other agents
8301	Serf LAN	TCP, UDP	Used by agents to process data flows within local network
8302	Serf WAN	TCP, UDP	Used by servers to exchange data with other servers via WAN
8400	CLI RPC	TCP	Used by all the agents to process RPC from CLI
8500	HTTP API	TCP	Used by clients to interact with HTTP API interface
8600	DNS	TCP, UDP	Used to allow DNS queries

4.11.5 consul.json configuration file and iwtm-consul.service unit file

Contents of **`consul.json`** configuration file

Content	Description
`{`	
`"bootstrap_expect": 1,`	Number of servers expected in cluster (for server mode only)
`"server": true,`	Start of agent in server mode (`false` – client mode)
`"datacenter": "iwtm",`	Title for datacenter

Content	Description
`"data_dir": "/opt/iw/tm5/var/consul",`	Directory to store the Consul service data
`"encrypt": "4RTZ5ttYY6RwIYX28XWNPw==",`	Secret key shared between cluster agents
`"log_level": "WARN",`	Level of logging
`"enable_syslog": true,`	Let logging in `syslog`
`"disable_update_check": true,`	Prohibit to check updates for Consul
`"leave_on_terminate": false,`	If true, then receiving SIGTERM, sends an exit message and leaves the cluster in standard mode. Default values: for server – `false`, for client – `true`
`"skip_leave_on_interrupt": true,`	If false, then receiving the signal of interruptions (e.g, SIGINT) leaves the cluster in standard mode. Default values: for server – `true`, for client – `false`
`"rejoin_after_leave": true,`	If `true`, then joins to cluster upon startup
`"limits": {`	Limit section
`"http_maxconns_per_client": 500`	Maximum number of HTTP(S) connections from one client
`},`	
`"retry_join": "0.0.0.0",`	In case of distributed installation contains the list of servers, which should be connected to
`"ui_config": {`	Web interface configuration section
`"enabled": false`	Access to web interface (prohibited by default)
`},`	
`"client_addr": "0.0.0.0"`	IP of a client with open web interface access
`}`	

Contents of **`iwtm-consul.service`** unit file

Code	Description
`[Unit]`	Section defining daemon metadata and its settings for interaction with other daemons
`Description=Consul Service discovery agent`	Daemon name
`Requires=network-online.target`	Services cannot operate without `network-online.target`
`After=network-online.service`	Daemons that will be run before the current daemon is started
`[Service]`	Daemon start setting section
`StandartOutput=syslog`	Redirecting standard output to `syslog`
`StandartError=syslog`	Redirecting standard error to `syslog`
`Type=simple`	Daemon start type. `simple` – daemon starts and switches to waiting mode on the console
`User=iwtm`	User name on behalf of which daemon is started
`Group=iwtm`	User group name
`SyslogIdentifier=iwtm-consul`	Defines process name for log prefix rows, which are sent to logging system or to core log buffer
`PermissionsStartOnly=true`	Starting commands from ExecStartPre on behalf of the **`root`** user
`ExecStartPre=/opt/iw/tm5/bin/consul validate /opt/iw/tm5/etc/consul`	Full path and command arguments which should be completed before starting the main process
`ExecStart=/opt/iw/tm5/bin/consul agent $OPTIONS -config -dir= /opt/iw/tm5/etc/consul`	Full path and command arguments which should be completed to start daemon. The command is to be run on behalf of the user defined in `User`
`ExecReload=/bin/kill -s SIGHUP $MAINPID`	The command restarts daemon configuration, if it is accessible. The command is to be run on behalf of the user defined in `User`

Code	Description
`KillSignal=SIGINT`	The command stops daemon's work forcefully
`TimeoutStopSec=5`	Period of time within which the system waits for daemon to stop, before labeling it as inaccessible or stopped forcefully
`[Install]`	Defining daemon's behaviour, if it is enabled or disabled
`WantedBy=multi-user.target`	Start this daemon, when the system is loading in a multi-user mode

4.12 Secure Deployment Recommendations

Traffic Monitor components use various network ports on the System servers.

> **Note:**
>
> For more information about component sets and templates on the System servers, see *InfoWatch Traffic Monitor. Installation guide. System Deployment Schemes and Installation Types*.
>
> For information about Traffic Monitor servers processes, see *Traffic Monitor Server Processes*.

To prevent unauthorized access to the components, it is recommended to configure a firewall. Only the connections that are necessary for the operation of the components should be allowed.

4.12.1 Components ports

Port	Interaction between components		Port's Function
	Source	Target	
Unique ports necessary for Traffic Monitor work			
TCP 9100	Device Monitor server	Traffic Monitor server	Transfer of shadow copies from Device Monitor for analysis
	Data Discovery server	Traffic Monitor server	Transfer of objects from Data Discovery for analysis

Port	Interaction between components		Port's Function
	Source	Target	
TCP 40731	Traffic Monitor Server with the "Interceptors" component	Data Storage server	Transfer of data for storing in Data Storage
	Traffic Monitor Server with the "Web-console" component	Data Storage server	Receiving data from Data Storage to access them in the web interface
TCP 30050	Traffic Monitor Server	Data Analysis Service server	Text data protoclustering using Data Analysis Service
TCP 9101	Server with a special connector	Traffic Monitor server	Transfer of objects for analysis using Push API Special connector using the Push API program interface can be created for transferring objects from servers via unsupported protocols, e.g. designed by client
	Lotus Domino adapter	Traffic Monitor server	Transfer of objects for analysis using Lotus Domino adapter
	MS Lync adapter	Traffic Monitor server	Transfer of objects for analysis using MS Lync adapter
	Smart Logger adapter	Traffic Monitor server	Transfer of objects for analysis using Smart Logger adapter
	MS ActiveSync adapter	Traffic Monitor server	Transfer of objects for analysis using ActiveSync adapter
Ports necessary for Traffic Monitor work in case of distributed installation			
TCP 1521	Traffic Monitor server	Traffic Monitor server DB	Connection to the Oracle Traffic Monitor database
TCP 5433	Traffic Monitor server	Traffic Monitor server DB	Connection to the PostgreSQL Traffic Monitor database
TCP 2025	Traffic Monitor server	Traffic Monitor server	SMTPD service

Port	Interaction between components		Port's Function
	Source	Target	
TCP 4301/2525/6338/6559	Traffic Monitor server	Traffic Monitor server	Sniffer service
TCP 8300	Traffic Monitor server	Traffic Monitor server	Consul (RPC) service
TCP 8301	Traffic Monitor server	Traffic Monitor server	Consul (Serf LAN) service
UDP 8301	Traffic Monitor server	Traffic Monitor server	Consul (Serf LAN) service
TCP 8302	Traffic Monitor server	Traffic Monitor server	Consul (Serf WAN) service
UDP 8302	Traffic Monitor server	Traffic Monitor server	Consul (Serf WAN) service
TCP 8400	Traffic Monitor server	Traffic Monitor server	Consul (CLI RPC) service
TCP 8500	Traffic Monitor server	Traffic Monitor server	Consul (HTTP API) service
TCP 8600	Traffic Monitor server	Traffic Monitor server	Consul (DNS interface) service
TCP 9997	Traffic Monitor server	Traffic Monitor server	Consul service connection to ICAP
TCP 9998	Traffic Monitor server	Traffic Monitor server	Consul service connection to Analysis
TCP 9999	Traffic Monitor server	Traffic Monitor server	Consul service connection to Messed
TCP 9090	Traffic Monitor server	Traffic Monitor server	Bookworm service
TCP 9091	Traffic Monitor management console	Traffic Monitor server	Sample Compiler service

Port	Interaction between components		Port's Function
	Source	Target	
TCP 9093	Traffic Monitor server	Traffic Monitor server	Blackboard service
TCP 9095	Traffic Monitor server	Traffic Monitor server	Licensed service
TCP 9096	Traffic Monitor server	Traffic Monitor server	Adlibitum service
TCP 9097	Traffic Monitor management console	Traffic Monitor server	System Check service
TCP 9099	Traffic Monitor server	Traffic Monitor server	Agent service
TCP 9102	Traffic Monitor server	Traffic Monitor server	Tech Tools service
TCP 9250	Traffic Monitor server	Traffic Monitor server	X2X service
TCP 9306	Traffic Monitor management console	Traffic Monitor server DB	Indexing service
TCP 9310	Traffic Monitor management console	Traffic Monitor server DB	Indexing service
TCP 9987	Traffic Monitor server	Traffic Monitor server	CAS service
TCP 9988	Traffic Monitor server	Traffic Monitor server	Luaengined service
TCP 9989	Traffic Monitor server	Traffic Monitor server	PAS service
TCP 16888	Traffic Monitor server	Traffic Monitor server	Qmover service
TCP 9103	Traffic Monitor server	Traffic Monitor server	Image_autoling service

Port	Interaction between components		Port's Function
	Source	Target	
TCP 9104	Traffic Monitor server	Traffic Monitor server	Text_autoling service
TCP 1111	Traffic Monitor server	Traffic Monitor server	Prometheus service (monitoring)
TCP 1112	Traffic Monitor server	Traffic Monitor server	Node-exporter service (monitoring)
TCP 1113	Traffic Monitor server	Traffic Monitor server	Consul-exporter service (monitoring)
TCP 1114	Traffic Monitor server	Traffic Monitor server	Postgres-exporter service (monitoring)
TCP 1115	Traffic Monitor server	Traffic Monitor server	Alert-manager service (monitoring)
TCP 1116	Traffic Monitor server	Traffic Monitor server	Statsd-exporter service (monitoring)
TCP 1117	Traffic Monitor server	Traffic Monitor server	Gearman-exporter service (monitoring)
TCP 1118	Traffic Monitor server	Traffic Monitor server	Process-exporter service (monitoring)
Standard ports for InfoWatch products interaction			
TCP 443	Device Monitor Server	Traffic Monitor server DB	Receiving Traffic Monitor policies
	Vision Server	Traffic Monitor server DB	Connection with Traffic Monitor DataExportAPI
	Activity Monitor Server	Traffic Monitor server DB	
	Prediction Server	Traffic Monitor server DB	
	Data Explorer Server	Traffic Monitor server DB	

Port	Interaction between components		Port's Function
	Source	Target	
TCP 5433	Traffic Monitor server	Remote DB server	The remote Traffic Monitor database in case of using it
Ports for integration with external services			
TCP 2025/587	Mail server	Traffic Monitor server	Simple Mail Transfer Protocol (SMTP)
	Traffic Monitor server	Mail server	
UDP 53	Traffic Monitor server	DNS server	Domain Name System (DNS)
UDP 123	Traffic Monitor server	NTP server	Network Time Protocol (NTP)
TCP 389	Traffic Monitor server	AD server	Lightweight Directory Access Protocol (LDAP)
	Device Monitor server	AD server	
TCP 636	Traffic Monitor server	AD server	Lightweight Directory Access Protocol (LDAPS)
TCP 3268	Traffic Monitor server	AD server	Microsoft Global Catalog
TCP 3269	Traffic Monitor server	AD server	Microsoft Global Catalog (LDAPS)
UDP 514	Traffic Monitor server	SIEM server	Integration with SIEM via the syslog protocol
TCP 1521	SIEM server	Traffic Monitor server DB	Integration with SIEM (Oracle Traffic Monitor DB)
TCP 5433	SIEM server	Traffic Monitor server DB	Integration with SIEM (PostgreSQL Traffic Monitor DB)
TCP 1344	Proxy server	Traffic Monitor server	Internet Content Adaptation Protocol (ICAP)

Port	Interaction between components		Port's Function
	Source	Target	
Remote management and administration			
TCP 22	Automated workplace	Traffic Monitor server	Configuring the server via SSH protocol
TCP 443	Automated workplace	Traffic Monitor Management console	Connection to Traffic Monitor Management console
TCP 3000	Automated workplace	Traffic Monitor server	Connection to the Grafana console (monitoring system)

4.13 Configuring Verdict Management

By default, you cannot manage verdict for Copy rules in Data protection policies, and value of **Assign verdict to event** field of **Action if rule triggered** block is set to *Allow*.

If you want the System to assign *Block* verdict to an event in case of a triggered rule, **enable verdict management**:

1. Connect to the database used by Traffic Monitor server.
2. Open `setting` table of the database.
3. Change the value of `allow_change_verdict_copy` parameter from `0` to `1`.

In Management Console, in the Copy rule for which you need to cofigure the verdict, you will be able to select a value of **Assign verdict to event** field: *Allow* or *Block.*

To disable verdict management, set 0 for `allow_change_verdict_copy` parameter.

> **Note:**
>
> In case you specify *Block* verdict and then disable verdict management in `setting` table, changing *Block* verdict for this rule wil be unavailable.

5 Settings for automatic deletion of Data Discovery events

Traffic Monitor applies a special anti-garbage filter to Data Discovery events. **By default**, the filter deletes Data Discovery events if they haven't triggered any policy or no protected objects have been found in them.

You can configure deletion of Data Discovery events in the `/opt/iw/tm5/etc/luaengined.conf` configuration file.

The anti-garbage filter modes are defined by the `DeleteDDEventsWithoutPolicy` parameter:

- `true`, to save only those events to the Traffic Monitor database that have triggered a policy,
- `false` (selected by default), to save only those events that contain protected objects or have triggered a policy.

If you do not want the anti-garbage filter to delete Data Discovery events, then in the `DisableDeleteDDEventsForServers` parameter, specify Data Discovery servers that should be added to exceptions.

To apply changes:

1. Save the file.
2. Restart the `iw_luaengined` service: `iwtm restart luaengined`

Example:
You do not want the anti-garbage filter to apply to events from MainDD and DDcompany servers. For other servers, you want to save only those events that contain triggered policies.

Parameter values for the luaengined.conf example configuration file:

```
"DeleteDDEventsWithoutPolicy": true,
"DisableDeleteDDEventsForServers": ["MainDD", "DDcompany"],
```

6 Monitoring

Monitoring subsystem exercises the following functions:

- Monitoring operations of all servers (both physical and virtual) included in InfoWatch Traffic Monitor solution;
- Monitoring of all software components (OS, DBMS, DB, processes etc.) included in InfoWatch Traffic Monitor solution;
- Control of indicator values for each server and component. Here by control we mean regular retrieval of indicator value and comparing this indicator value to a threshold value;
- Enabling and disabling monitoring of specific indicators and servers;
- Sending email notifications when indicator values exceed the normal range.

The Monitoring subsystem is autoinstalled during InfoWatch Traffic Monitor server installation by the installer. For more details on the installation see *InfoWatch Traffic Monitor. Installation and Configuration Guide* document.

If InfoWatch Traffic Monitor system is deployed on few servers using the installer and installation templates, the monitoring subsystem will be automatically configured and detectors will be in accordancy with the templates used.

You can also use *Configuring Indicators Manually*.

6.1 Monitoring Subsystem for Distributed Architecture

> **Warning!**
>
> If InfoWatch Traffic Monitor system is deployed on few servers using installer and installation templates, the monitoring subsystem will be automatically configured in accordance with installation templates used.

Monitoring configuration includes the following tasks:

- Configuring Connection to Device Monitor;
- Configuring Indicators Manually;
- Configuring Addresses of Time Synchronization Server for Monitoring Subsystem;
- Configuring Notifications.

6.1.1 Configuring Connection to Device Monitor

To configure connection to Device Monitor server:

1. Edit `/opt/iw/tm5/etc/nagios/iwmon/iwmon-hosts-dm.cfg` file:
   - Uncomment `host` and `hostgroup` sections.
   - In `address` parameter of `host` section define host name or IP-address of Device Monitor server.
- In `/opt/iw/tm5/etc/nagios/iwmon/iwmon-services-dm.cfg` file uncomment `service` section.
- In `/opt/iw/tm5/etc/nagios/iwmon/iwmon-commands.cfg` file uncomment `Check Dm server` section.
- Restart nagios process:
  `systemctl restart iwtm-nagios`

6.1.2 Configuring Indicators Manually

Enabling and disabling indicators is being performed in the following file of the `/opt/iw/tm5 /etc/nagios/iwmon` directory:

- iwmon-services-db-psql.cfg – Postgre SQL indicators configuration;
- iwmon-services-db.cfg – Oracle indicators configuration;
- iwmon-services-dm.cfg – Device Monitor indicators configuration;
- iwmon-services-loadavg.cfg – Server load parameters configuration (System load indicator);
- iwmon-services.cfg – main indicator configuration;
- iwmon-services-queue.cfg – queues indicators configuration.

The service_description parameter displays each indicator purpose.

To enable indicator, change `register` parameter value to `1`.

To enable indicator, change `register` parameter value to `0`.

To apply changes, restart nagios:

```
systemctl restart iwtm-nagios
```

6.1.3 Configuring Addresses of Time Synchronization Server for Monitoring Subsystem

To provide correct time synchronization, IP address of NTP server should be specified at every system server. You may use any exact time service utilizing NTP protocol and available from your network: both network equipment and Windows domain controllers.

To configure time synchronization at the server:

1. Open `/opt/iw/tm5/etc/nagios/iwmon/iwmon-services-ntp.cfg` file for editing.
2. In the check command parameter replace default IP-address with the actual IP-address of NTP server (time synchronization server).

6.1.4 Configuring Notifications

The monitoring subsystem enables sending email notifications where the current indicator value exceeds the threshold value.

For e-mail notifications to be sent only for selected indicators:

1. Go to the Nagios subsystem directory:
   `cd /opt/iw/tm5/etc/nagios/iwmon`
2. Open the **iwmon-services.cfg** configuration file:
   `mcedit /opt/iw/tm5/etc/nagios/iwmon/iwmon-services.cfg:`
3. To identify the needed indicator, view the `service_description` field.
4. In the indicator section, set the `notifications_enabled` parameter to:
   - 1 – to enable sending notifications;
   - 0 – to disable sending notifications.
     If there is no `notifications_enabled` parameter in the configuration file, add this parameter manually.

- Save changes.
- Set up a connection to one of the mail servers:
  - the built-in local Traffic Monitor mail server with anonymous access (Posfix or Exim) OR

- an external mail server without encryption.

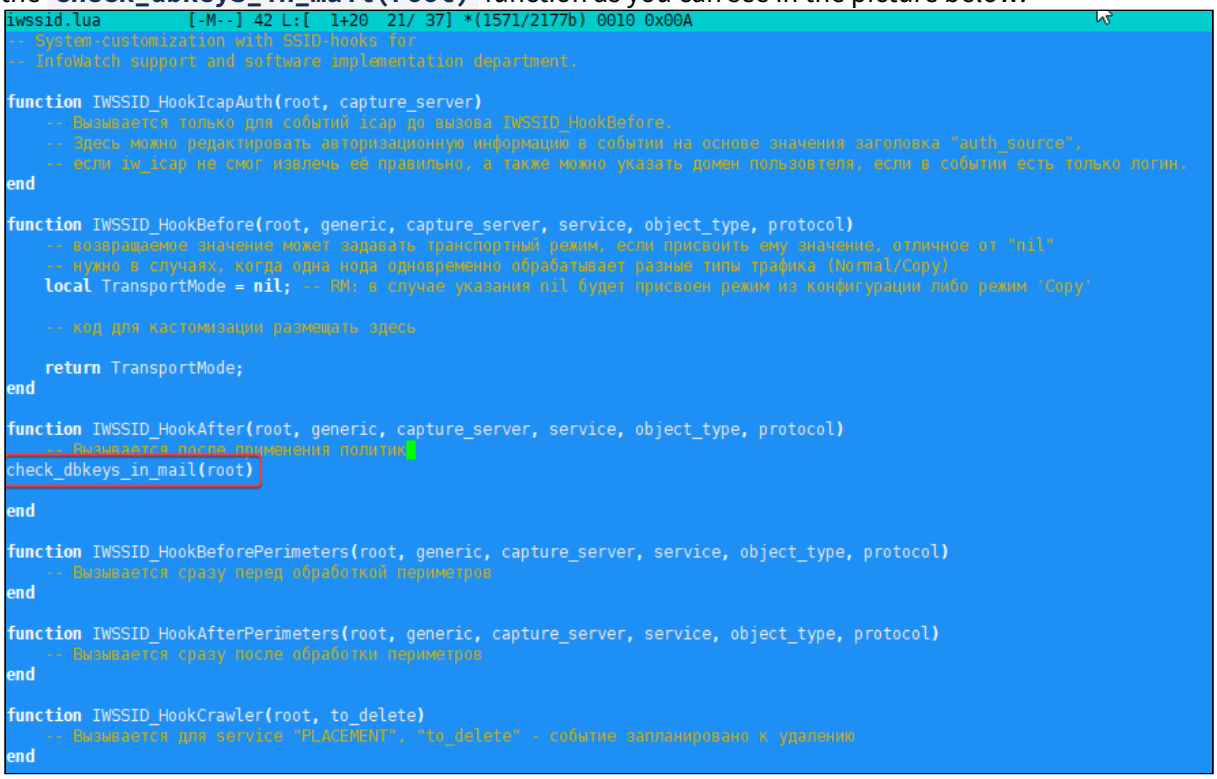
7 Using a database unloading for automatic recipient validataion

Traffic Monitor can use a database unloading to control sending data by email. The database unloading must meet the requirements (see *InfoWatch Traffic Monitor. User Guide. Recommendations on Preparing Unloadings*). The last column of each row in the unloading must contain the email address to which data of this row is allowed to be sent. The fields of the last column are considered keys and the unloading is not indexed by them. This solution allows you to protect a large amount of data without manually creating a lot of protected objects and policies. In the *Blocking* transport mode (the in-line mode), Traffic Monitor allows you to block sending emails with protected data or send them to quarantine.

By default, after Traffic Monitor installation, this functionality is disabled.

To enable the ability to use the database unloading for recipient validation:

1. In the `/opt/iw/tm5/etc/scripts/iwssid.lua` file, add a command to call the `check_dbkeys_in_mail(root)` function as you can see in the picture below.



2. After you save changes, check permissions with the command:

```
ls -la /opt/iw/tm5/etc/scripts/iwssid.lua
```

Example where the correct permissions are displayed:

```
-rw-r--r--. 1 iwtm iwtm 2181 jul 24 09 : 43 /opt/iw/tm5/etc/scripts/iwssid.lua
```

3. If the permissions have been changed and do not match the example above, use the following command to edit the permissions:

```
chown -R iwtm:iwtm /opt/iw/tm5/etc/scripts/iwssid.lua
```

4. Restart Traffic Monitor services with the command:

```
iwtm restart
```

Important!

Please, keep in mind that parameters of the `cas.conf` configuration file affect the operation with database unloadings:

- `MinWords` (`10` by default). The System creates an event with a violation if an email with an impermissible recipient contains at least 10 words, including words which satisfy detection conditions. Otherwise, the System does not detect a protected object and creates an event without a violation;
- `MaxCheckLimit` (`0` by default). If the value is greater than zero, the System continues searching until it finds the specified maximum number of lines with unique keys (or number of unique lines in case the unloading does not contain keys). The parameter is available only for *All words in one cell following each other in any order* search type and if optimization if disabled (`"StopOnLimitExceed": false`).

For details, see *InfoWatch Traffic Monitor. Configuration Files Guide*.

To disable the functionality:

1. In the `/opt/iw/tm5/etc/scripts/`**`iwssid.lua`** file, delete the command that calls the **`check_dbkeys_in_mail(root)`** function.
2. Restart Traffic Monitor services with the command:

```
iwtm restart
```

8 Database Administration

This chapter contains information on administrating:

- Oracle;
- PostgreSQL.

DB statistics information is available in *Compiling DB statistics* KB article.

8.1 Oracle

This chapter contains instructions on Oracle administration:

- Tablespaces in InfoWatch Traffic Monitor database;
- Managing Daily Tablespaces;
- Load statistics of Daily Tablespaces;
- Database Backup;
- Database Routinely Maintenance;
- Database Logging.

> **Note:**
>
> To change the DB user password, see the *Changing the Preset Password* article.

8.1.1 Tablespaces in InfoWatch Traffic Monitor database

General information

InfoWatch Traffic Monitor data are stored in DB as tablespaces (TS). There are two types of Traffic Monitor tablespaces:

The type of tablespace	Purpose
Main	Storing the settings that are needed for the analysis and processing of the objects (configuration, tags, colors, etc.). Managing the System using the Management Console (roles, user accounts, etc.)
Daily	Storing objects intercepted within one day. Storing information on the results of the analysis and processing of the objects (analyzed object categories, terms, etc.)

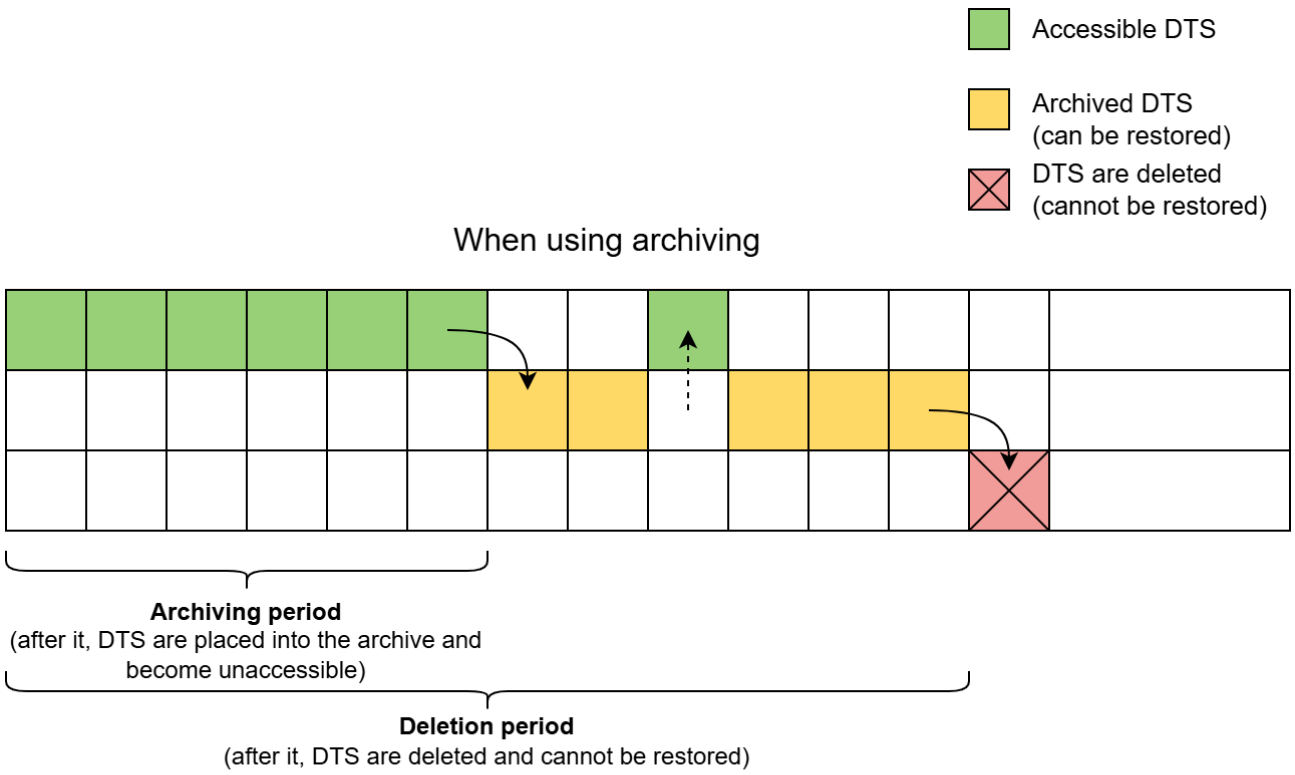
All data about the objects intercepted on a particular day (according to UTC+0) is stored in three daily tablespace (DTS) of the folowing types:

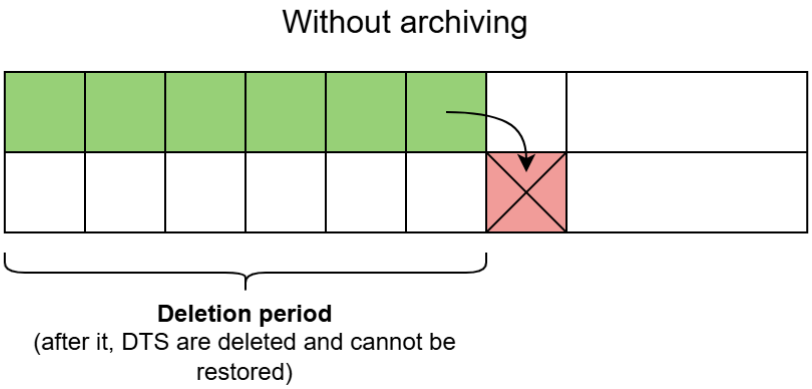
- Events without violations;
- Events with violations;
- Other (screenshots).

Daily tablespaces are created every day in such a way that the database always has DTS for the next six days, excluding the current day. All DTS automatically get the *IWTM_YYYYMMDD_X* name, where X is the DTS number (the TBS_ID value from the TBS_LIST table). The tablespace type is defined by the TBS_TYPE value: 0 – No Violations, 1-Violation, 2 - Other (screenshots).

Lifecycle of DTS can be divided into 3 stages:

1. **Ordinary storing**
   Initially, DTS are stored in the database and are in operational access. If you use Data Storage, then the event contents from this DTS such as attachments, screenshots, audio files, and so on, are stored there.
2. **Storing in an archive** (optional)
   If you need to store events for a long time, but do not need operational access to them, you can archive them before deleting.
   For this, enable automatic archiving of DTS. After the **archiving period**, they will be archived and will become unavailable. If neccesary, you can restore them (the dashed arrow on the diagram).
3. **Deleting**
   To avoid disk overflow, you can delete DTS which you do not need to store.
   For this, enable automatic deleting of DTS. After the **deletion period**, they will be deleted without the possibility of restoring.





By default, automatic archiving and deletion are disabled. You can enable them during the installation or after it. Also, you should set the DTS **archiving period** and/or the DTS **deletion period**. For details, see articles on archiving and deletion.
Configure **archiving period** and **deletion period** so that you always have enough free space on the disks, which store DTS, and they do not overflow.

If you need to archive/restore/delete a single DTS, you can do it manually in the Traffic Monitor web interface. See *InfoWatch Traffic Monitor. User Guide. Managing tablespaces*.
If the web interface is unavailable, use the **iw_space_manager** service utility for this (see *Managing Daily Tablespaces with the Console Utility*). Last operate with DTS manually in the database.

Storing

The main tablespace is stored in the `/u02/oradata` catalogue by default.
The daily tablespaces are stored in the `/u02/oradata1` catalogue by default.
For details on placing DTS, see *Configuring Location of Files within File System*.

On storage modes, see *Configuring Storage Modes for Tablespace Files*.

Archiving

Archiving process consists of:

- disconnecting DTS from the database;
- moving DTS to the specified catalogue (by default: `/u02/arch`).

You should place this catalogue on the separate disk to free up space on the hard drive with operational data and avoid its overflow.
When using Data Storage, binary data of the archived DTS will be moved to the Data Storage slow disk.

During archiving, data is not compressed. So, on the allocated disk, it will take about the same place. You can estimate the size of the disk for archived DTS based on how much place DTS take in operational access in their storage directories (by default: `/u02/oradata1` in the DB and `/opt /dsdata/_seaweedfs` in Data Storage).

Archiving is described in details in the *Archiving Daily Tablespaces* section.

To access the archived DTS data, you need to restore it. See *Restoring Daily Tablespaces*.

Deleting

When automatic deletion is enabled, the System deletes DTS older than the deletion period. You specify its value separately for each DTS type: events without violations, events with violations, other (screenshots).

For details on deletion methods, see *Deleting Daily Tablespaces*.

8.1.2 Managing Daily Tablespaces

In this section:

- Configuring Location of Files within File System;
- Configuring Storage Modes for Tablespace Files;
- Archiving Daily Tablespaces;
- Restoring Daily Tablespaces;
- Deleting Daily Tablespaces;
- Managing Daily Tablespaces with the Console Utility.

Configuring Location of Files within File System

By default, the installation program (see *InfoWatch Traffic Monitor. Installation Guide*) sets the usage of one directory for storing daily tablespaces: `/u02/oradata1`.

> **Note:**
>
> For tablespace archives, the `/u02/arch` directory is used.

However at high loads, it is recommended to place daily TS on different I/O devices (LUN, physical drives) for sequential usage. For example, when 3 file systems are defined, data is located as follows:

First day – the daily tablespace is created in File system 1.

Second day – the daily tablespace is created in File system 2.

Third day – the daily tablespace is created in File system 3.

Fourth day – the daily tablespace is created in File system 1.

…

Distribution of daily TS files (number and locations) of daily TS files can be managed using scripts.

An example of the script to change the number of dedicated daily TS storage locations:

```
BEGIN
pkg_part.set_df_path_cnt('4');
COMMIT;
END;
/
```

> **Warning!**
>
> After changing the number of daily TS storage locations make sure to adjust (add/delete) their location paths.

An example of the script to change the location paths for daily TS:

```
BEGIN
pkg_part.set_df_path('/test1/', 1);
pkg_part.set_df_path('/test2/', 2);
pkg_part.set_df_path('/test3/', 3);
pkg_part.set_df_path('/test4/', 4);
COMMIT;
END;
/
```

> **Note:**
>
> It is advisable to add “/” symbol in the end whenever specifying the path.

An example of viewing daily TS content:

```
select a.tbs_name, a.service_code, power(10, trunc(log(10, a.text_size))) ||
'-' ||
power(10, trunc(log(10, a.text_size)) + 1) size_range,
count(1) cnt, sum(a.text_size) text_size
from
(
select t.tbs_name, o.service_code,
case
when length(o.text) > 0 then length(o.text)
else 1
end text_size
from object o
```

```
inner join object_source os on os.object_id = o.object_id
inner join tbs_list t on o.tbs_id = t.tbs_id
) a
group by a.tbs_name, a.service_code, power(10, trunc(log(10, a.text_size))) ||
'-' ||
power(10, trunc(log(10, a.text_size)) + 1)
order by 1, 2, 3
```

Service code definition is specified in **services.xml** file, which is located in `/opt/iw/tm5/etc/compendiums/` catalogue.

> **Note:**
>
> To receive intercepted objects volume analysis along with DB service information, use SQLDeveloper

Configuring Storage Modes for Tablespace Files

There are three modes of storing data in DB:

- **Normal** – switching to the following partition (when it is specified) happens after the previous one is overflow.
- **Fast/Slow disks** – dividing pools into fast and slow ones. Recent data are to be kept in fast pool. In a number of days specified, the System relocate them to the slow ones. Slow pool operates in Normal mode.
- **Rotate** – switching to the following partition happens every day once the previous one is overflow.

You select storage mode when installing Traffic Monitor. By default, the installer offers you **Normal** mode. You can change the storage mode later.

By default, DTS are stored in the `/u02/oradata1` directory.

> **Note:**
>
> Modes described in this article relate to data stored in DB only. Changing mode does not affect data stored in Data Storage.

Fast/slow disks Mode

The System moves data from the fast disk to the slow one every night. If you are switching from **Normal** mode to **Fast/slow** mode, you store a large amount of DTS on your old partition, and you specify the old partition as a fast disk, then data transfer may take a long time.
If you will use the old disk as a fast one, before switching mode make sure that the following conditions are met:

- The new partition size is not less than the old one size. So, there is enough space for transferring all DTS from the old partition;
- There is free space in the old partition not less than the largest tablespace that will be transferred to the slow disk.

The System creates DTSs every day in such a way that the database always has DTS for the next six days, excluding the current day. So, if when switching from **Normal** mode to **Fast/slow** mode, you specify a new partition as a fast disk, then, for first seven days, the System will save events to the already created DTS on the slow disk. In this case, the System will create new DTSs on the fast disk.

For configuring **Fast/slow** mode, we will use the new patrtition `/u03/oradata/` as a slow disk and the old partition `/u02/oradata1/` as a fast one.

To configure "Fast/slow" storage mode (fast and slow drives):

1. Edit the `/etc/fstab` file to automatically mount a new partition to `/u03:`
   `/dev/sdd1 /u03/ ext4 defaults 1 2`
2. Create the `oradata` folder:
   `mkdir /u03/oradata`
3. Change user:
   `chown -R oracle:oinstall /u03`
4. Log into the database:
   `su - oracle`
   `sqlplus iwtm@iwtm`
5. Specify the storage period on fast drives:
   `exec pkg_part.set_fast_days(7);`
6. Specify the path to the fast drive:
   `exec pkg_part.set_fast_path('/u02/oradata1/',1);`
7. Specify the path to the slow drive:
   `exec pkg_part.set_df_path('/u03/oradata/',1);`
8. If needed, you can use several slow drives. The data will be recorded to the next slow drive when the previous is full. To add the slow drives specify a new path:
   `exec pkg_part.set_df_path('/u03/oradata2/',2);`
9. Specify the total amount of slow drives:
   `exec pkg_part.set_df_path_cnt('2') ;`
10. Change the mode to fast-slow drives:
    `exec pkg_part.set_filesys_type('fast/slow');`
    The data will be transferred from fast drives to slow ones during night job.
11. Apply changes:
    `commit;`
12. To run the process of moving files manually:
    `exec iwtm.pkg_part.move_fast_tablespaces_to_slow();`

> **Note:**
>
> Few slow drives can be used in the System.

> **Note:**
>
> In **Fast/slow** mode, the screenshot tablespaces are stored in slow partitions only.

Rotate Mode

To configure "Rotate" storage mode (everyday rotation):

1. Mount the partition and make sure that it can be mounted automatically on reboot
2. Create the `oradata` folder:
   `mkdir /u03/oradata`
3. Change user:
   `chown -R oracle:oinstall /u03`

4. Log into the database:
   ```
   su - oracle
   sqlplus iwtm@iwtm
   ```
5. Add a path to the second partition:
   ```
   exec pkg_part.set_df_path('/u03/oradata/',2);
   ```
6. Specify a new number of paths:
   ```
   exec pkg_part.set_df_path_cnt(2);
   ```
7. Apply changes:
   ```
   commit;
   ```
8. Switch to **rotate** mode:
   ```
   exec pkg_part.set_filesys_type('rotate');
   ```

   ```
   SQL>select * from setting where setting like 'df_path%';
   select * from setting where setting like 'df_filesys%';

   setting | value | editable
   ------------+---------------+----------
   df_path1 | /u02/oradata1/ | 1
   df_path2 | /u03/oradata/ | 1
   df_path_cnt | 2 | 1
   df_filesys_type | rotate | 1
   ```

> **Note:**
>
> Parameters of modes are stored in the `setting` table of the database. Their specifics are described in the Knowledge Base article *Tablespaces data storage settings. Data storage partition*.

Archiving Daily Tablespaces

Archiving process consists of disconnecting DTS from the database and moving them to the specified catalogue (by default: `/u02/arch`). You should place this catalogue on the separate disk to free up space on the hard drive with operational data and avoid its overflow. You can restore this data if necessary.

In the System, there are following ways of archiving DTS:

- Automatically after the specified period is expired (see *Automatic Archiving of Daily Tablespaces*);
- Manual archiving of single DTS:
  - In the Traffic Monitor web interface. See *InfoWatch Traffic Monitor. User Guide. Managing tablespaces*;
  - Using the console utility of the **iw_space_manager** service (see *Managing Daily Tablespaces with the Console Utility*);
  - In the database (see *Manual Archiving of Daily Tablespaces in the Database*).

If you need to archive a single DTS, we recommend to do it in the Traffic Monitor web interface. If the web interface is unavailable, use the **iw_space_manager** service utility. Last use manual archiving of DTS in the database.

Archiving when using Data Storage

If you use Data Storage, then, when archiving DTS, binary data will be moved to the Data Storage slow disk.
If the slow disk is not connected, then, in Traffic Monitor web interface, archived daily tablespace will have the "*Archived*" status. But in this case, binary data of DTS will be stored on the fast disk of the Data Storage server,

inside the directory for placing binary data. You specify this directory when installing the storage (by default: `/opt/dsdata/_seaweedfs`).

You specify the path to the slow disk when installing Data Storage. For details, see *InfoWatch Traffic Monitor. Installation Guide. Installing Platform*.
To add the path to the Data Storage slow disk after installing the storage:

1. Create a directory and mount the slow disk to it:

   ```
   mkdir -p /mnt/slowdisk
   mount /dev/<your_slowdisk> /mnt/slowdisk
   ```

2. Set a user and access permissions for the directory, where you have mounted the slow disk:

   ```
   chown 1000 /mnt/slowdisk
   chmod 700 /mnt/slowdisk
   ```

3. Open the Data Storage deployment for editing. For this use the command:

   ```
   kubectl -n infowatch edit deployment datastorage-<node_label>
   ```

   where `<node_label>` (hereafter) is the label of the node where the storage is installed. For the central node or, if the storage is installed not in the cluster of Platform products, use the label `central`. For the office node, the label is specified during installation (`office` by default).
4. Add the following lines to the `containers` -> `volumeMounts` section:

   ```
       - mountPath: /var/lib/dataslow
         name: datastorage-dataslow-host-volume-<node_label>
   ```

5. Add the following lines to the `volumes` section:

   ```
       - hostPath:
           path: /mnt/slowdisk
           type: ""
         name: datastorage-dataslow-host-volume-<node_label>
   ```

6. Save the changes and exit the editor.

Automatic Archiving of Daily Tablespaces

By default, automatic DTS archiving is disabled. To make the System archive DTS according to schedule, enable this function and set the time after which DTS must be archived. Details are described further.
Every day, at 00:00 server time, the **iw_space_manager** service procedure will archive DTS that satisfy the following condition:

The date of a tablespace is less or equal to the difference between the current date and the specified time interval for archiving the tablespace.

So, all DTS older than the specified value will be archived.

Warning!

If months or years are used as the time interval, we recommend using the maximum value of the interval in calculations. For a month – 31 days. For a year – 366 days. For example, to archive DTS older than 4 years, specify the interval 366 * 4 = 1464 days.

Before starting automatic archiving, check, that archiving catalogue has been set properly:

```
select value from setting where setting = 'archive_path';
```

The `Oracle` user should be an owner of the catalogue.

In case the catalogue has been set improperly, reset it with the following command:

```
begin
sp_setting_set('archive_path', '/u02/arch/');
commit;
end;
/
```

To perform TS archiving automatically (by default the function is disabled) you can use the following scripts (run under the account of the database schema owner):

- For daily TS storing objects with *Violation* status:

```
begin
sp_setting_set('violation_archive_enabled', 1);
sp_setting_set('violation_archive_period', D);
commit;
end;
/
```

- For daily TS storing objects with *NoViolation* status:

```
begin
sp_setting_set('noviolation_archive_enabled', 1);
sp_setting_set('noviolation_archive_period', D);
commit;
end;
/
```

- For daily TS storing *screenshots*:

```
begin
sp_setting_set('other_archive_enabled', 1);
sp_setting_set('other_archive_period', D);
commit;
end;
```

```
/
```

where:
1 – indicates that automatic archiving is enabled (to disable, specify 0 instead);
D – is a number of days at the end of which the daily TS will be archived by the System. This number should be smaller than the set number of days before deleting the daily tablespace (see *Deleting Daily Tablespaces*).

Manual Archiving of Daily Tablespaces in the Database

It is recommended to archive single DTS via web interface or using the **iw_space_manager** utility.

The order of archiving daily tablespaces manually in the database is described below.

> **Important!**
>
> In this method of archiving, data stored in the Data Storage are not archived.

Selecting daily tablespaces and data files

To get a list of disconnected tablespaces, you must run the following query to the database under the account of the database schema owner. The queries are built according to the archiving policy of your company. For example, to select the daily TS storing the objects which were intercepted more than 100 days ago, run the following query:

```
SELECT t.tbs_id, t.tbs_name, t.part_date

FROM tbs_list t

WHERE t.part_date<trunc(SYSDATE) - 100 + 23 / 24 AND t.status IN (0, 3)
```

> **Warning!**
>
> It is strongly recommended to run the following sample query in this section in order to retrieve the list of files. This is due to the fact that the file list if generated by any other ways may be incomplete.

In this case, the following query is created to select data files:

```
SELECT f.file_name

FROM dba_data_files f

WHERE f.tablespace_name in

(

  SELECT t.tbs_name

  FROM tbs_list t

  WHERE t.part_date<trunc(SYSDATE) - 100 + 23 / 24 AND t.status IN (0, 3)
```

```
)
```

where `t.status IN (0, 3)` indicates that the tablespaces with the *ReadWrite* (status code `0`) and *ReadOnly* (status code `3`) statuses are selected for archiving.

Disconnecting Daily Tablespaces from the Database

> **Warning!**
>
> It is strongly recommended not to disable daily TS when the IWTM_ADD_PARTS, IWTM_DELETE_TABLESPACES and IWTM_ARCHIVE_TABLESPACES jobs are running as it may cause the data loss.

1. Call the procedure on behalf of the owner of the database schema, call the procedure:

   ```
   begin

   IWTM.pkg_part.archive_tablespace(N);

   end;
   /
   ```

   where `N` is a value of tbs_id attribute of the target TS from the `tbs_list` table `.`
   After running this script status of daily TS will change to *Offline*.
2. Repeat the procedure call for each daily TS you want to disable.

Moving Datafiles

Transfer the data files belonging to the disabled daily TS to the other data carrier.

To get a list of data files for all disconnected tablespaces, use the following script:

```
select tbs_name from tbs_list where NOTE = 'Tablespace is offline'
```

where `tbs_name` is a name of a tablespace.

`tbs_list` is a name of a table.

Data files are stored in the catalogue:

```
select value from setting where setting = 'archive_path'.
```

Restoring Daily Tablespaces

> **Warning!**
>
> A tablespace can only be restored in the database schema, from which it was disconnected (even if this schema was updated). The restoration of the tablespace after total database schema reinstallation is impossible.

In the System, there are the following ways of restoring DTS:

- In the Traffic Monitor web interface. See *InfoWatch Traffic Monitor. User Guide. Managing tablespaces*;
- Using the console utility of the **iw_space_manager** service. See *Managing Daily Tablespaces with the Console Utility*;
- In the database – this method is described below. It is recommended to use this method last.

Moving data files

To restore daily TS, you must move data files of this tablespace from a data carrier to the folder which path you can obtain using the following command:

```
select value from setting where setting = 'archive_path';
```

Warning!

Make sure that the *oracle* user has reading and writing privileges for the directory to move the data files to.

Connecting the daily tablespace

Warning!

Do not connect daily TS during IWADDPARTS, IWTM_DELETE_TABLESPACES jobs. This may cause the data loss.

1. On behalf of the owner of the database schema, call the procedure:

   ```
   begin
   pkg_part.restore_tablespace(N);
   end;
   /
   ```

   where `N` is ID of the tablespace (specified in tbs_list).
   After procedure ending the status of TS will change to *Restored*.
2. Repeat the procedure call for each daily TS you want to enable.

Deleting Daily Tablespaces

In the System, there are the following ways of deleting DTS:

- Automatic, according to schedule. Described below.
- Manual deleting of single DTS:
  - In the Traffic Monitor web interface. See *InfoWatch Traffic Monitor. User Guide. Managing tablespaces*;
  - Using the console utility of the **iw_space_manager** service. See *Managing Daily Tablespaces with the Console Utility*;
  - In the database. Described below.

We recommend using scheduled deletion. Otherwise, disk overflow can occur or the big amount of data for deletion can build up. It will result in increasing the time for running this procedure in future and consequently increasing the idle time of the Traffic Monitor Server.

> **Warning!**
>
> Deleting tablespaces cannot be undone. If you complete the operation, you will not be able to restore the deleted data.
> The System deletes both DTS data from the storage directory (by default, `/u02/oradata1`, or `/u02/arch` for the archived ones) and information on DTS from the database. If you use Data Storage, data from it will be deleted as well.
>
> While deleting tablespaces, access to the corresponding tables is closed. For this reason, log files of the **iw_deliver**, **iw_x2db**, **iw_updater** processes may contain database access errors. Once tablespaces are deleted, these processes automatically restore access to the database.

Automatic deletion

By default, automatic DTS deletion is disabled. To make the System delete DTS according to schedule, enable this function and set the time after which DTS must be deleted. Details are described further.
Every day, at 00:00 server time, the **iw_space_manager** service procedure will delete DTS that satisfy the following condition:

The date of a tablespace is less or equal to the difference between the current date and the specified time interval for deleting the tablespace.

So, all DTS older than the specified value will be deleted.

> **Warning!**
>
> If months or years are used as a time interval, then we recommend you to apply the maximal interval value in the counting: 31 days for a month and 366 days for a year. For example, to delete DTS older than 4 years, specify 366*4=1464 days interval.

Enabling automatic deletion

Automatic deletion according to schedule is configured separately for each DTS type.
To enable it, set the deletion `delete_enabled` flag and specify the period of storing DTS `delete_period`.
For this, run the following scripts on behalf of the database scheme owner:

- For DTS storing objects with the *Violation* status:

```
begin
sp_setting_set('violation_delete_enabled', 1);
sp_setting_set('violation_delete_period', D);
commit;
end;
/
```

- For daily TS storing objects with the *No Violation* status:

```
begin
sp_setting_set('noviolation_delete_enabled', 1);
sp_setting_set('noviolation_delete_period', D);
commit;
end;
/
```

- For daily TS storing *screenshots*:

```
begin
sp_setting_set('other_delete_enabled', 1);
sp_setting_set('other_delete_period', D);
commit;
end;
/
```

where:
1 indicates that automatic deletion is enabled (to disable, specify 0 instead of 1);
D is a number of days at the end of which the DTS will be deleted by the System. This number should be larger than the set number of days before archiving daily tablespaces (see *Automatic Archiving of Daily Tablespaces*).

Deleting daily tablespaces with a script

As well, you can configure automatic deletion of daily tablespaces, use the **/opt/iw/tm5/bin/dbtools/dbconf-iwdrop-oracle.sh** script. You can use the following commands:

Purpose	Command
Set up deletion period	`/opt/iw/tm5/bin/dbconf-iwdrop-oracle.sh set violation|noviolation|other <days> [-v]`
Enable automatic deletion	`/opt/iw/tm5/bin/dbconf-iwdrop-oracle.sh enable violation|noviolation|other [-v]`
Disable automatic deletion	`/opt/iw/tm5/bin/dbconf-iwdrop-oracle.sh disable violation|noviolation|other [-v]`
View the state of settings	`/opt/iw/tm5/bin/dbconf-iwdrop-oracle.sh show [-v]`

Example:

To enable automatic deletion of the DTS, that store objects with the *"No violation"* status and are older than 15 days, run the commands:

```
/opt/iw/tm5/bin/dbconf-iwdrop-oracle.sh set noviolation 15 [-v]
/opt/iw/tm5/bin/dbconf-iwdrop-oracle.sh enable noviolation [-v]
```

Manual deletion in the database

If you need to immediately launch the procedure of automatic DTS deletion (e.g., due to file system space shortage the interval of deletion was changed, but the next launch of the job will not happen soon), run the following script on behalf of the database scheme owner:

```
begin
     iwtm.pkg_part.delete_tablespaces();
end;
/
```

As a result, the System will delete DTS for which scheduled deletion is enabled and the period of storing is specified.

The command above affects data in DB only. If you use Data Storage, launch the procedure of automatic DTS deletion using the **iw_space_manager** service:

```
iw_space_manager cron
```

To delete one DTS by its identifier (ID), run the following script:

```
begin
     iwtm.pkg_part.delete_tablespace(ID);
end;
/
```

This command affects data in DB only. If you use Data Storage, delete DTS using the **iw_space_manager** service:

```
iw_space_manager tbs:delete <tbs_id>
```

Managing Daily Tablespaces with the Console Utility

To manage daily tablespaces via the command line, use the console utility that calls the **iw_space_manager** service. The service is available for the **iwtm** user and is enabled by default.

When using Data Storage, the **iw_space_manager** service archives/restores/deletes binary data from the storage as well.

Commands and parameters of the service:

Command	Description
`tbs:archive [options] [--] <id>`	Starts archiving a daily tablespace with its ID specified. An example on how to archive a daily tablespace with ID=123: `iw_space_manager tbs:archive 123`

Command	Description
`tbs:create <type> <capture>`	Creates a daily tablespace with the following data specified: • type: 0 – no violation, 1 – violation, 2 – screenshots • creation date in the YYYY-MM-DD format An example on how to create a new daily tablespace with violations on September 1, 2023: `iw_space_manager tbs:create 1 2023-09-01`
`tbs:delete [options] [--] <id>`	Deletes a daily tablespace with its ID specified
`tbs:get [options]`	Gets the list of existing daily tablespaces. Additional parameters: • `--limit=LIMIT` – sets the maximal number of items to display on the screen • `--offset=OFFSET` – hides the first N items (N=0 by default) • `--order=ORDER` – sorts the daily tablespace list by: ASC – daily tablespace name, DESC – daily tablespace creation date An example on how to get a list of 20 daily tablespaces sorted by the creation date: `iw_space_manager tbs:get -- limit=20 --order=DESC`
`tbs:restore [options] [--] <id>`	Restores an archived daily tablespace with its ID specified. An example on how to restore a daily tablespace with ID=123 from an archive: `iw_space_manager tbs:restore 123`
	With all the commands listed above, you can use the following common parameters: • `-f, --force` – runs the command without the validation check in case of a database emergency. Enables you to perform operations on tablespaces where date of event interception coincides with today's date; • `--used-configs` – runs the command with the parameters from the configuration file; • `-p, --configpath [=CONFIGPATH]` – displays the configuration file path. By default, it is `opt/iw/tm5/etc`; • `--force-config-load` – force loads the configuration file even if the query parameters are specified.

To get logs and debug messages, run the service with the parameters below:

- `-v` – displays usual logs;
- `-vv` – displays logs with extended detalization;
- `-vvv` – displays logs with full detalization.

To get a list of available commands, run the command:

`iw_space_manager list`

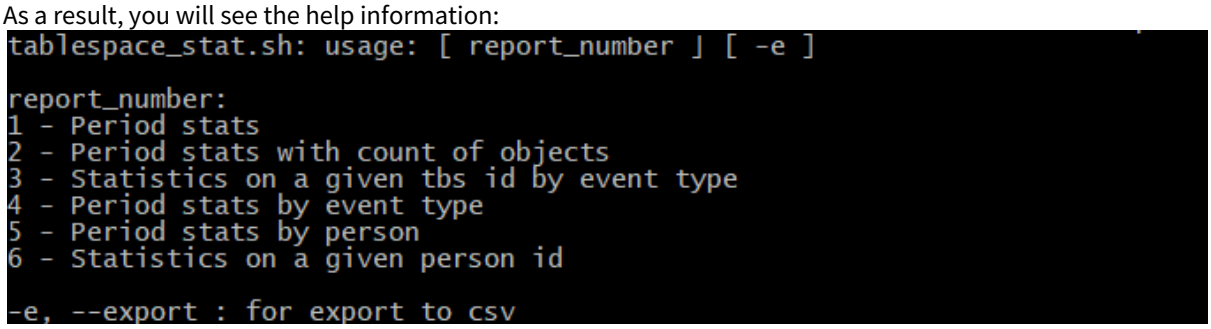
8.1.3 Load statistics of Daily Tablespaces

To analyze a load of daily tablespaces, you can use the `/opt/iw/tm5/bin/dbtools/tablespace_stat_oracle/tablespace_stat_oracle.sh` script. With its help, you can get the statistics on TBS content for the selected period. The script executes one of the six scenarios and in each case it collects different information. The scenarios are described in more detail *below*.

To display the information on the script usage:

1. Go to the directory where the script is located by running the command:
   `cd /opt/iw/tm5/bin/dbtools/tablespace_stat_oracle/`
2. Run the script without parameters by running the command:
   `./tablespace_stat_oracle.sh`

As a result, you will see the help information:



To use the script:

1. Go to the directory where the script is located by running the command:
   `cd /opt/iw/tm5/bin/dbtools/tablespace_stat_oracle/`
2. Run the script by running the command:
   `./tablespace_stat_oracle.sh <scenario_number> <-e>`
   where the scenario number is a digit from 1 to 6;
   `-e` or `--export` is a key that you can specify to unload the results of script execution in .csv format. The results will be displayed in the command line window, as well.
   Example of a command running the script:
   `./tablespace_stat_oracle.sh 1`
3. Specify the parameters depending on a scenario:
   - For scenarios 1, 2, 4, 5, 6 – a period for which the statistics should be displayed:
     a. Specify the lower bound of the period in the dd.mm.yyyy format and press **Enter**;
     b. Specify the upper bound and press **Enter**;

     > **Note:**
     >
     > If you do not specify values for bounds and just press **Enter** twice then, by default, statistics for last two weeks will be displayed.
     > If you specify only the lower bound then information for the period from the specified date to the current one will be displayed.

   - For scenario 3 – ID of the tablespace;
   - For scenario 6 – person's ID.

In case of unloading the results of script performing, the file is saved in the directory where the script is located. The file has the following name:

`report_<scenario_name>_<date>.csv`
where `<scenario_name>` has a value depending on the scenario number:

Scenario number	Scenario name	Short description
1	tbs_stat_period	Statistics for the period of time
2	tbs_stat_period_count	Statistics for the period of time including count of events in TBS
3	tbs_stat_by_tbs_id	Statistics on a particular TBS
4	tbs_stat_by_event_type	Statistics on event types for the period
5	tbs_stat_by_persons	Statistics on top-10 persons
6	tbs_stat_by_person_id	Statistics on a particular person

`<date>` - date of script execution. It has the `<yyyy-mm-dd_HH-mm-ss>` format.
Example of the unloading file name: `report_tbs_stat_period_count_2024-12-17_18-20-39.csv`

Scenarios of script execution

Scenario 1

As a result of scenario 1 execution, statistics on space that tablespaces take up in DB for the period of time will be displayed.

```
Period tablespace stats

Date format: dd.mm.yyyy (empty - no filter. Default - 14 days)
Enter value for start_date: 27.11.2024
Enter value for end_date: 28.11.2024
not spooling currently

  TBS_TYPE | PART_DATE   | TBS_SIZE           | AVG_TBS_SIZE
---------- | ---------- | ------------------ | ------------------
         2 | 27.11.2024 |            2.00 mb |          124.17 mb
         1 | 27.11.2024 |            2.00 mb |          124.17 mb
         0 | 27.11.2024 |            2.00 mb |          124.17 mb
         2 | 28.11.2024 |          140.00 mb |          124.17 mb
         1 | 28.11.2024 |          255.00 mb |          124.17 mb
         0 | 28.11.2024 |          344.00 mb |          124.17 mb

6 rows selected.
```

The following information will be shown in columns:

TBS_TYPE	TBS type: 0 - No Violations; 1 - Violation; 2 - Other (screenshots).

PART_DATE	Date of TBS creation
TBS_SIZE	TBS size
AVG_TBS_SIZE	Average TBS size for the specified period of time

Scenario 2

As a result of scenario 2 execution, more detailed statistics on tablespaces for the period of time will be displayed.

```
Period tablespace stats with count of objects

Date format: dd.mm.yyyy (empty - no filter. Default - 14 days)
Enter value for start_date: 27.11.2024
Enter value for end_date: 28.11.2024
not spooling currently

    TBS_ID | PART_DATE  |   TBS_TYPE | TBS_STATUS    | COUNT_OBJECTS | TBS_SIZE
---------- | ---------- | ---------- | --------------| ------------- | ------------------
        69 | 27.11.2024 |          2 | 0 - online    |             0 |           2.00 mb
        68 | 27.11.2024 |          1 | 0 - online    |             0 |           2.00 mb
        67 | 27.11.2024 |          0 | 0 - online    |             0 |           2.00 mb
        72 | 28.11.2024 |          2 | 0 - online    |           104 |         140.00 mb
        71 | 28.11.2024 |          1 | 0 - online    |           312 |         255.00 mb
        70 | 28.11.2024 |          0 | 0 - online    |           424 |         344.00 mb

6 rows selected.
```

The following information will be shown in columns:

TBS_ID	TBS ID
PART_DATE	Date of TBS creation
TBS_TYPE	TBS type: 0 - No Violations; 1 - Violation; 2 - Other (screenshots).
TBS_STATUS	TBS status: 0 - online (connected, in normal state); 3 - read only (only for reading); 5 - before exchange (before disconnecting for archiving); 7 - prepared offline (prepared to be disconnected for archiving); 9 - disconnected (disconnected before moving to archive); 10 - offline (archived and disconnected); 11 - plugged (plugged in before restoration); 13 - before update (before updating for restoration); 15 - after update (after updating for restoration); 17 - before revert (before restoring from archive); 20 - restored (restored from archive).
COUNT_OBJECTS	Count of events in a TBS

TBS_SIZE	TBS size

Scenario 3

As a result of scenario 3 execution, statistics on a particular TBS will be displayed.

```
Statistics on a given tbs id by event type

Enter value for tbs_id: 70
not spooling currently

TYPE_NAME                    | TYPE_SIZE
-----------------------------|------------------
Cloud storage                |          43.41 mb
Print                        |          32.21 mb
WhatsApp                     |          30.25 mb
Email                        |          16.01 mb
```

The following information will be shown in columns:

TYPE_NAME	Event type
TYPE_SIZE	Overall size of the events of the particular type in TBS

Scenario 4

As a result of scenario 4 execution, statistics on event types for the specified period of time will be displayed.

```
Period stats by event type

Date format: dd.mm.yyyy (empty - no filter. Default - 14 days)
Enter value for start_date: 21.11.2024
Enter value for end_date: 30.11.2024
not spooling currently

OBJECT_TYPE                  | OBJECTS_SIZE       | COUNT_OBJECTS | MAX_OBJECTS_IN_HOUR
-----------------------------|--------------------|---------------|-------------------
Email                        |          43.41 mb  |            86 |                 40
Cloud storage                |          32.21 mb  |            65 |                 45
Print                        |          30.25 mb  |            39 |                 27
WhatsApp                     |          16.01 mb  |            26 |                 18
```

The following information will be shown in columns:

OBJECT_TYPE	Event type
OBJECTS_SIZE	Overall size of the events of the particular type
COUNT_OBJECTS	Count of the events of the particular type
MAX_OBJECTS_IN_HOUR	Maximum count of events per hour during the day

Scenario 5

As a result of scenario 5 execution, statistics on top-10 persons for the specified period of time will be displayed. Events with participation of top-10 persons take up the largest space in DB.

```
Period stats by persons

Date format: dd.mm.yyyy (empty - no filter. Default - 14 days)
Enter value for start_date:
Enter value for end_date:
not spooling currently

PERSON_ID                                | DISPLAY_NAME                  | OBJECT_COUNT | OBJECT_SIZE
---------------------------------------- | ----------------------------- | ------------ | ------------------
B37174C262444412B7AC8ECDCA544BD200000000 | Ivanov Ivan                   |            3 |            0.24 mb
099914C8CFD6416CB734C7721A2CF37600000000 | Petrov Petr                   |            2 |            0.62 mb
554BEE71D78B49CE80DA30301022AC9400000000 | Margaret Smith                |            1 |            0.47 mb
```

The following information will be shown in columns:

PERSON_ID	Person ID
DISPLAY_NAME	Person display name
OBJECT_COUNT	Count of events
OBJECTS_SIZE	Overall size of the events with participation of the person

Scenario 6

As a result of scenario 6 execution, statistics on particular person for the specified period of time will be displayed.

```
Statistics on a given person id

Enter value for person_id: B37174C262444412B7AC8ECDCA544BD200000000
Date format: dd.mm.yyyy (empty - no filter. Default - 14 days)
Enter value for start_date:
Enter value for end_date:

Ivanov Ivan

OBJECT_TYPE                  | OBJECT_COUNT | OBJECTS_SIZE
---------------------------- | ------------ | ------------------
Электронная почта            |            3 |            0.24 mb
```

The following information will be shown in columns:

OBJECT_TYPE	Event type
OBJECT_COUNT	Count of events
OBJECTS_SIZE	Overall size of the events with participation of the person

8.1.4 Database Backup

To minimize the risk of data loss, we strongly recommend to backup the database on a monthly basis. To store backups, we recommend you to use dedicated storage system.

Experienced Oracle database administrators can perform a backup at the running database (“hot backup”) using Oracle-recommended RMAN program (see Oracle documentation), previously taken the Oracle database to ARCHIVELOG mode.
For non-database administrators who do not have standard backuping methods we recommend creating a “cold backup” at the stopped database. You may also use this procedure to migrate the database from one server to another.

If, in addition to database, you use Data Storage, i.e. a storage of binary data, then back up the storage together with DB. This procedure is decribed together with DB backup creation.

This section further covers:

- Creating Database Backup;
- Restoring the Database from the Backup.

Creating Database Backup

Creating database backup (cold backup procedure) is performed in the following order:

- Defining the Size of Backup;
- Checking the Backup Storage;
- Creating Backup Directories;
- Creating Recovery Aid Files;
- Collecting a List of Files to Backup;
- Stopping IWTM System;
- Stopping Oracle Database;
- Copying Database Files to Backup Storage;
- Creating Data Storage Backup.

Defining the Size of Backup

To define size of the future archive, it is necessary to find out summary size of catalogues of DB and Data Storage, if use you the storage.

To define the DB size:

1. On the computer, where the Oracle DB is installed enter the System as the **oracle** user.
2. Open Oracle SQL*Plus and log as **sysdba**:
   `OS> sqlplus / as sysdba`
3. If the system was installed using the installer, you can use **wsqlplus** command.

   > **Note:**
   >
   > `wsqlplus` command is an alias of `sqlplus` command:
   > `alias wsqlplus='rlwrap -b "" -f ~/sql.dict sqlplus'`

   Wait until the connection is established - you will receive **Connected** message.
4. Copy (or enter) the following SQL query to the command prompt:
   `SQL> SELECT ROUND(SUM(bytes)/1024/1024/1024,4) GB FROM ( SELECT SUM(bytes) bytes FROM dba_data_files union SELECT SUM(bytes) bytes FROM dba_temp_files union SELECT SUM(bytes) bytes FROM v$log);`
5. Write down the received database size.
6. Exit Oracle SQL*Plus:
   `SQL> exit`

To define size of data in the storage:

1. On the Data Storage server, enter the system as the **root** user.
2. Get the `/opt/dsdata/_seaweedfs` catalogue size:
   `du -sx -BM /opt/dsdata/_seaweedfs`

Checking the Backup Storage

To check if the selected backup storage is suitable, take the following steps:

1. Mount the external storage to the DB server. Make sure that the following requirements are met:
   - The external storage, the database and Data Storage are located on different computers.
   - Computers with servers and databases to be backed up can access the external storage.
   - The external storage has more free space on the hard disk than the backup size.
2. Define the amount of free space on the hard disk:
   a. Run the following command as **root**:
      `OS>df -h`
   b. Make sure that the mounted section has more free space than the backup size.

Creating Backup Directories

> **Warning!**
>
> Backup directories should be located at computer other than the database is located.

1. Log the system as **root** user.
2. Create the directory for backup files storage:
   `OS> mkdir /opt/IWTM_Backup_Files`
3. Create subdirectories:
   ```
   OS> mkdir /opt/IWTM_Backup_Files/Oradata
   OS> mkdir /opt/IWTM_Backup_Files/IWTM
   OS> mkdir /opt/IWTM_Backup_Files/Recovery_Aid
   ```
4. Grant permissions to these directories to the **oracle** user:
   `OS> chmod 777 /opt/IWTM_Backup_Files/ -R`

Creating Recovery Aid Files

To provide database restoring, besides data files, you need recovery aid files: a **trace** file of the control file and a copy of the **init.ora** file.

The **trace** file of the control file contains the names and locations of all of the data files and additional data files that have been added to the database. It also contains the redo logs and the commands that can be used to recreate the database structure.

The **init.ora** file contains Oracle initializing parameters, including the names and locations of the database control files.

To create trace file of the control file, spfile file and to backup them:

1. Log on to the computer that hosts the Oracle database as the **oracle** user.
2. Open Oracle SQL*Plus and log as **sysdba** user:
   `OS> sqlplus / as sysdba`
   If the system was installed using the installer, you can use **wsqlplus** command.

   > **Note:**

`wsqlplus` command is an alias of `sqlplus`:

```
alias wsqlplus='rlwrap -b "" -f ~/sql.dict sqlplus'
```

Wait for **Connected** message.

3. Copy (or enter) the following SQL queue to the command prompt:
   `SQL> alter database backup controlfile to trace;`
4. To find the directory in which the trace file was created, enter:
   `SQL> show parameter user_dump;`
   At that folder you can find **trc** file and **alert_iwtm.log** file (alert_INSTANCE_NAME). At log file you can find the name of the trace file which was created.
   Exit Oracle SQL*Plus:
   `SQL> exit`
5. If the system was installed using installer and default parameters were left, you can check last **alert_iwtm.log** records by the following command:
   `OS> tail -f /u01/app/oracle/diag/rdbms/iwtm/iwtm/trace/alert_iwtm.log`
   Press CTRL+C to exit view.
6. Create a copy of **spfile** (**pfile**):
   `OS> sqlplus / as sysdba`
   `SQL> CREATE PFILE from SPFILE;`
7. To find the directory in which the pfile file was created, enter:
   `SQL> show parameter PFILE`
   It returns a path to the directory where pfile was created (/u01/app/oracle/product/db_1/dbs/ by default).
   In this folder the **initiwtm.ora** file is located too. Please create a backup for **initiwtm.ora** file.
   Exit Oracle SQL*Plus:
   `SQL> exit`
8. Copy trace and spfile files to the backup storage, to previously created `/opt/IWTM_Backup_Files/Recovery_Aid` subdirectory.
9. Be sure to copy the file with the most recent date and timestamp. To check the date and the timestamps of the files in the directory, enter:
   `OS> ll`
10. Rename the file so that it can be easily identified, for example:
    `controlfilebackupMMDDYY.trc`

Collecting a List of Files to Backup

Create a list of files that need to be backed up. You will use this list in a later step.

1. Log on to the computer that hosts the Oracle database as the **oracle** user.
2. Open Oracle SQL*Plus and log as **sysdba** user:
   OS> sqlplus / as sysdba
   Wait for **Connected** message.
3. Create the list of directories to backup using following SQL commands:
   `SQL> SELECT file_name FROM dba_data_files UNION SELECT file_name FROM dba_temp_files UNION SELECT name FROM v$controlfile union select member name from v$logfile;`
   The output is like:
   `/u01/app/oracle/fast_recovery_area/iwtm/control02.ctl`
   `/u01/app/oracle/oradata/iwtm/control01.ctl`
   `/u01/app/oracle/oradata/iwtm/redo01.log`

```
/u01/app/oracle/oradata/iwtm/redo02.log
/u01/app/oracle/oradata/iwtm/redo03.log
/u01/app/oracle/oradata/iwtm/redo04.log
/u01/app/oracle/oradata/iwtm/redo05.log
/u01/app/oracle/oradata/iwtm/redo06.log
/u01/app/oracle/oradata/iwtm/redo07.log
/u01/app/oracle/oradata/iwtm/redo08.log
/u01/app/oracle/oradata/iwtm/redo09.log
/u01/app/oracle/oradata/iwtm/redo10.log
/u01/app/oracle/oradata/iwtm/redo11.log
/u01/app/oracle/oradata/iwtm/redo12.log
/u01/app/oracle/oradata/iwtm/redo13.log
/u01/app/oracle/oradata/iwtm/sysaux01.dbf
/u01/app/oracle/oradata/iwtm/system01.dbf
/u01/app/oracle/oradata/iwtm/temp01.dbf
/u01/app/oracle/oradata/iwtm/undotbs01.dbf
/u01/app/oracle/oradata/iwtm/users01.dbf
/u02/oradata/IWTM_2012_06_25_1.dbf
/u02/oradata/IWTM_2012_06_26_1.dbf
/u02/oradata/IWTM_2012_06_27_1.dbf
/u02/oradata/IWTM_2012_06_28_1.dbf
/u02/oradata/IWTM_2012_06_29_1.dbf
/u02/oradata/IWTM_2012_06_30_1.dbf
/u02/oradata/IWTM_2012_07_01_1.dbf
/u02/oradata/IWTM_2012_07_02_1.dbf
/u02/oradata/IWTM_2012_07_03_1.dbf
/u02/oradata/IWTM_2012_07_04_1.dbf
/u02/oradata/iwtm_1.dbf
```

4. Exit Oracle SQL*Plus:
   ```
   SQL> exit
   ```

Stopping IWTM System

Stopping IWTM system is not required but we recommend doing this especially if there is little free space on IWTM server.

To stop IWTM system:

1. On each computer that hosts IWTM services, log in as **root**.
2. Stop all running Traffic Monitor Server services:
   ```
   iwtm stop
   ```
3. You can start services later (when backup is finished) using the following command:
   ```
   iwtm start
   ```

Stopping Oracle Database

> **Waning!**
>
> Please stop ORACLE database before backuping database files.

To stop Oracle database:

1. On the computer that hosts the database, log on as the ***oracle*** user.
2. In the command prompt, enter:
   `OS> dbshut $ORACLE_HOME`
3. You can start services later (when backuping finished) using the following command:
   `OS> dbstart $ORACLE_HOME`

Copying Database Files to Backup Storage

To copy database files:

1. Stop all Oracle services (see *Stopping Oracle Database*). To make sure that all services are stopped, run the following command:
   `ps aux | grep ora`
   If Oracle services are not stopped, backup files can be damaged and unsuitable for recovery.
2. On the computer that hosts the database, copy data from DB directories (see *Collecting a List of Files to Backup*) to previously created directories. You need to copy all the contents of the directories including links.
   If the system was installed using the installer and default parameters were left, the path should be as follows:
   a. copy contents of the **/u01/app/oracle/oradata/iwtm/** directory to the **/opt/IWTM_Backup_Files/IWTM** directory
   b. copy contents of the **/u02/oradata/** directory to the **/opt/IWTM_Backup_Files/Oradata** directory
   c. copy the **/u01/app/oracle/product/db_1/dbs/orapwiwtm** file to the **/opt/IWTM_Backup_Files/Recovery_Aid** directory

Important!

Use only the following devices as file storages:

- an external drive with Ext4 or XFS file system
- a remote block device connected via iSCSI or NFS protocol
- a locally connected block device with Ext4 or XFS file system

Creating Data Storage Backup

In Data Storage, Traffic Monitor data are stored in buckets. To back up the storage of binary data, you need to back up these buckets. This procedure consists of the following steps:

1. Enabling functionality of backup
2. Getting a token to work with the Data Storage service
3. Getting a list of buckets for backup
4. Creating the backup task
5. Checking successful completion of backup

During the backuping process you need to send requests using the **curl** utility. When doing this, you can specify the following values as a server address `<ADDRESS>` :

- **localhost:40731**, if you send requests from the server where Data Storage is installed.
- **<server_IP_address>:40732**, if you send requests from another server.

Enabling functionality of backup

To enable creating a backup of Data Storage, you need to specify a directory for that.

We recommend to do it when you install the storage. For this, run the installer using the `--dsbackupspath` parameter with the directory for backups.
Example: `./setup.py install --dsbackupspath=/opt/dsbackups`

To specify the directory after installation, add it to the Data Storage deployment. For this:

1. On the Data Storage server, create a directory for storing a backup:

   ```
   mkdir /opt/dsbackups
   ```

2. Mount an external storage for backups into this directory. Requirements for the backup storage are described in the *Checking the Backup Storage* article.
3. Set a user and access permissions for the created directory:

   ```
   chown 1000 /opt/dsbackups
   chmod 700 /opt/dsbackups
   ```

4. Open the Data Storage deployment for editing. For this use the command:

   ```
   kubectl -n infowatch edit deployment datastorage-<node_label>
   ```

   where `<node_label>` (hereafter) is the label of the node where the storage is installed. For the central node or, if the storage is installed not in the cluster of Platform products, use the label `central`. For the office node, the label is specified during installation (`office` by default). The following actions depends on a text editor that you use. In our example, we use **vi**. To enter insert mode, press **i**.
5. Add the following lines to the `containers` -> `volumeMounts` section:

   ```
   - name: datastorage-databackups-host-volume-<node_label>
     mountPath: /var/lib/databackups
     mountPropagation: HostToContainer
   ```

   In this and the next step, indent with spaces and do not use a tab.
6. Add the following lines to the `volumes` section:

   ```
   - name: datastorage-databackups-host-volume-<node_label>
     hostPath:
       path: /opt/dsbackups
   ```

7. Save the changes and exit the editor.
   For example, if you use the **vi** editor, press **Esc** and type `:wq`.

Getting a token to work with the Data Storage service

To work with the Data Storage service, you need to get a Bearer token for HTTP authorization. In requests to Data Storage, specify it in the Authorization header.

The token value can become outdated. So in response to a request, the service returns the 440 HTTP code. In this case, get the token again.

To get the token, send the following request to Data Storage:

```
curl -X GET -I --silent "http://<ADDRESS>/v2/logon?extAPIToken=<DS_user_token>" 2>&1 | grep
'^Authorization'
```

where `<ADDRESS>` is the server address. For details, see above.
`<DS_user_token>` is the Data Storage authorization token. The default value: `1exwo92iwpczl1hy6jxy`.
To learn more, see *Configuring Connection with Data Storage*.

As a result, you will see the Authorization header in the following format:
`Authorization: Bearer <token_value>`

Getting a list of buckets for backup

To get a list of identifiers of Data Storage buckets which store Traffic Monitor data, connect to the Traffic Monitor database and run the query:

```
select distinct(storage_bucket_id) from iwtm.tbs_list;
```

If you need to know a list of all existing Data Storage buckets (their bucketID identifiers), send the following request to the storage service:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets?filter=*" 2>&1
```

where `<AUTH_HEADER>` is the Authorization header received on the previous step;
`<ADDRESS>` is the server address. For details, see above.

Results in the JSON format may look as follows:


```
{
  "data": [
    {
      "backendType": "seaweedfs",
      "id": "barbucket",
      "isDumped": false,
      "isInaccurate": false,
      "isSlowDrive": false,
      "size": 27632
    },
    {
      "backendType": "seaweedfs",
      "id": "bazbucket",
      "isDumped": false,
      "isInaccurate": false,
      "isSlowDrive": false,
      "size": 27633
    },
    {
      "backendType": "file",
      "id": "oldbucket",
      "isDumped": false,
```

```
      "isInaccurate": false,
      "isSlowDrive": false,
      "size": 27634
    }
  ]
}
```

Also you can filter the bucket list with a regular expression using the **jq** utility. For example, the following request will find all buckets whose bucket ID value begins with the "bar" substring:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets?filter=*" 2>&1 | jq --raw-output
'.data.[] | select(.backendType=="seaweedfs") | select(.id|test("^bar")).id'
```

Creating the backup task

To create a task of bucket backup, send the following request:

```
curl -X POST --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/backups?bucketID=<BUCKET_ID>"
```

where `<AUTH_HEADER>` is the Authorization header;
`<ADDRESS>` is the server address. For details, see above;
`<BUCKET_ID>` is the bucket identifier.

In response, the ID of the created task will be displayed:

```
{
  "taskID": "fe8163f2-a0fb-4990-8bb7-87aee5c425bc"
}
```

Also you can create a bucket backup task and get the ID of this task using the **jq** utility:

```
curl -X POST --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/backups?bucketID=<BUCKET_ID>" |
jq --raw-output '.taskID'
```

Next, you can check successful completion of bucket backup using the task ID.

Checking successful completion of backup

To make sure that backuping completed successfully, check the status of the task. For this, send the request:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/backups?taskID=<TASK_ID>"
```

where `<AUTH_HEADER>` is the Authorization header;
`<ADDRESS>` is the server address. For details, see above;
`<TASK_ID>` is the task identifier.

If succesfull, you will see names of data and index files created in the backup directory:

```
{
  "bucketID": "foobucket",
```

```
  "dataFilename": "foobucket_0.tar.gz",
  "indexFilename": "foobucket_extindex.tar.gz",
  "taskStatus": "Success"
}
```

In case of failure, the error description will be displayed:

```
{
  "bucketID": "foobucket",
  "error": "permission denied: cannot extract data tarball",
  "taskStatus": "Success"
}
```

Make sure that the task is completed and has one of the following statuses: Success, Canceled, Failed.

Also you can get the task status using the **jq** utility:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/backups?taskID=<TASK_ID>" | jq
--raw-output '.taskStatus'
```

Restoring the Database from the Backup

To restore the database and/or Data Storage:

1. Choose the appropriate method of database recovery if necessary:
    - If the database was damaged due to the system failure or user error, restore the old database. For example, if an important file has been occasionally deleted, you can restore the database to the state when this file still existed. For details, see *Restoring at the Same Database*.
    - If the old database cannot be used any more, create a new database and recover data to it. See *Restoring at the New Database*.
2. Follow one of the scenarios below. You can find required instructions in the followinf articles: *Stopping IWTM System*, *Stopping Oracle Database* and *Restoring Data Storage*.

If you use Data Storage and it is not damaged then no action with the storage is required when restoring DB. Data Storage does not interact with the database directly.

To restore only DB:

1. Stop the Traffic Monitor services.
2. Restore the DB the chosen way.
3. Start the Traffic Monitor services.

To restore only Data Storage:

1. Stop the Traffic Monitor services.
2. Stop the database.
3. Restore Data Storage.
4. Start the database.
5. Start the Traffic Monitor services.
6. Make sure that Traffic Monitor is connected with Data Storage. If necessary, configure connection. For detailes, *Configuring Connection with Data Storage*.

To restore both DB and Data Storage:

1. Stop the Traffic Monitor services.
2. Restore Data Storage.
3. Restore the database.
4. Start the Traffic Monitor services.
5. Make sure that Traffic Monitor is connected with Data Storage. If necessary, configure connection.

> **Important!**
>
> If you need to update Traffic Monitor to a new version, you should first restore the database. Updating Traffic Monitor either before or during the database recovery is not supported.

Restoring at the Same Database

Below you can find instructions on how to restore data to a database that has the same directory structure as the one that was backed up.

To restore the database without creating a new one:

1. Make sure the database environment is healthy. Check the existing database, the database server that hosts the existing database, and the computer that hosts the database server.
2. Stop database:
   ```
   OS> dbshut $ORACLE_HOME
   ```
3. Delete all content of the **/u02/oradata/, /u02/oradata1/**, and **/u01/app/oracle/oradata/iwtm/** directories.
4. Copy the content of the **/opt/IWTM_Backup_Files/Oradata/** directory to the **/u02/** directory.
5. Copy the content of the **/opt/IWTM_Backup_Files/IWTM/** directory to the **/u01/app/oracle/oradata/iwtm/** directory.
6. Rename the **control01.ctl** file located in the **/opt/IWTM_Backup_Files/IWTM/** directory to the **control02.ctl** and copy it to the **/u01/app/oracle/oradata/iwtm/** directory.
7. Check permissions. Change them if necessary:
   ```
   OS> chown -R oracle:oinstall /u02/
   OS> chown -R oracle:oinstall /u01/app/oracle/oradata/iwtm/
   ```
8. Start database:
   ```
   OS> dbstart $ORACLE_HOME
   ```

Restoring at the New Database

If the drive structure of the new database differs from the drive structure of the old database, perform the following actions:

1. Edit the **initiwtm.ora** file in the **/opt/IWTM_Backup_Files/Recovery_Aid** directory to reflect the drive structure of the new database.
2. Copy the edited **initiwtm.ora** file to the **$ORACLE_HOME/dbs** directory on the computer that hosts the new database.
3. Log on to the computer that hosts the Oracle database as the **oracle** user.
4. Open Oracle SQL*Plus and connect as the **sysdba** user:
   ```
   OS> sqlplus / as sysdba
   ```
5. Generate spfile
   ```
   SQL> create spfile from pfile;
   ```
6. If you need to rename some files, you can use the following script:
   ```
   OS> sqlplus / as sysdba
   SQL> startup mount
   ```

```
SQL> alter database rename file '/<path_to_proddb_files>/<filename1>' to '/
<path_to_clonedb_files>/<filename1>';
…
SQL> alter database rename file '/<path_to_proddb_files>/<filenameN>' to '/
<path_to_clonedb_files>/<filenameN>';
SQL> shutdown immediate
```

Here `path_to_proddb_files` is a path to the directory where new database files are located (to where backup is restoring), and `path_to_clonedb_files` is a path to the directory where backup files are located.

Restoring Data Storage

Restoring the storage of binary data lies in restoring buckets from their backups. This procedure consists of the following steps:

1. Getting a token to work with the Data Storage service
2. Creating the task of restoring bucket
3. Checking successful completion of the task

When sending **curl** requests, you can specify the following values as a server address `<ADDRESS>`:

- **localhost:40731**, if you send requests from the server where Data Storage is installed.
- **<server_IP_address>:40732**, if you send requests from another server.

Getting a token to work with the Data Storage service

To work with the Data Storage service, you need to get a Bearer token for HTTP authorization. In requests to Data Storage, specify it in the Authorization header.
The token value can become outdated. So in response to a request, the service returns the 440 HTTP code. In this case, get the token again.

To get the token, send the following request to Data Storage:

```
curl -X GET -I --silent "http://<ADDRESS>/v2/logon?extAPIToken=<DS_user_token>" 2>&1 | grep
'^Authorization'
```

where `<ADDRESS>` is the server address. For details, see above.
`<DS_user_token>` is the Data Storage authorization token. The default value: `1exwo92iwpczl1hy6jxy`.
To learn more, see *Configuring Connection with Data Storage*.

As a result, you will see the Authorization header in the following format:
`Authorization: Bearer <token_value>`

Creating the task of restoring bucket

To create a task of restoring bucket, send the following request:

```
curl -X POST --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/restores?bucketID=<BUCKET_ID>"
```

where `<AUTH_HEADER>` is the Authorization header;
`<ADDRESS>` is the server address;
`<BUCKET_ID>` is the bucket identifier.

In response, the ID of the created task will be displayed:

```
{
  "taskID": "fe8163f2-a0fb-4990-8bb7-87aee5c425bc"
}
```

Also you can create the task of restoring bucket and get the ID of this task using the **jq** utility:

```
curl -X POST --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/restores?bucketID=<BUCKET_ID>"
 | jq --raw-output '.taskID'
```

Next, you can check successful restoration of bucket using the task ID.

> **Important!**
>
> You can restore the bucket only if there is no one with the same ID in Data Storage.

Checking successful completion of the task

To make sure that restoration completed successfully, check the status of the task. For this, send the request:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/restores?taskID=<TASK_ID>"
```

where `<AUTH_HEADER>` is the Authorization header;
`<ADDRESS>` is the server address.;
`<TASK_ID>` is the task identifier.

If succesfull, you will see the following response:

```
{
  "bucketID": "foobucket",
  "taskStatus": "Success"
}
```

Make sure that the task is completed and has one of the following statuses: Success, Canceled, Failed.

Also you can get the task status using the **jq** utility:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/restores?taskID=<TASK_ID>" | jq
--raw-output '.taskStatus'
```

8.1.5 Database Routinely Maintenance

> **Warning!**
>
> Do not power DB server off with power button: in certain cases this may cause the DB corruption.

When doing routine database maintenance, you should follow the established order:

1. Close all instances of the Management Console. Make sure that all connections of other programs to DB Traffic Monitor schema are closed; if there're any active connections, disable them.
2. Stop all TM processes on Traffic Monitor server:
   ```
   iwtm stop
   service php-fpm stop
   service nginx stop
   ```
3. Retrieve list of scheduled jobs from database server. For thisб run the following query in sqlplus using schema owner account:
   ```
   select JOB_NAME, STATE
   from user_scheduler_jobs u
   where u.enabled = 'TRUE'
   ```
4. Disable jobs by running the following script:
   ```
   BEGIN
   dbms_scheduler.disable('JOB_1',true);
   …
   dbms_scheduler.disable('JOB_N',true);
   COMMIT;
   END;
   /
   ```
   where `JOB_1… JOB_N` are the names of tasks to disable.
5. To make sure that none of these jobs is running run the following query:
   ```
   SELECT job_name
   FROM user_scheduler_running_jobs
   A list of active jobs displays. If any of the jobs is running you may run the query to stop it:
   BEGIN
   dbms_scheduler.stop_job('job_name');
   COMMIT;
   END;
   /
   ```
6. To prevent hanging on, stop Device Monitor server:
   ```
   net stop iwdms
   ```
7. Make all the necessary operations with the database.
8. After all required operations are made check connection to database server from Traffic Monitor server:
   ```
   sqlplus db_login/db_password@tns_name
   ```
   Here `db_login` and `db_password` are the name and password of the database schema owner, and `tns_name` is the name of TNS service.
   If the test is successful, the following SQL *Plus prompt will be displayed in response to the command:
   SQL>
9. Run Device Monitor server:
   ```
   net start iwdms
   ```
10. Run the processes of the Traffic Monitor server:
    ```
    iwtm stop
    service php-fpm stop
    service nginx stop
    ```
11. Check if the system log contains errors. The path to the system log:
    ```
    /var/log/messages
    ```
12. If the system log contains any error messages, consult technical support service.

13. Enable running previously disabled jobs:
    ```
    BEGIN
    dbms_scheduler.enable('JOB_1',true);
    …
    dbms_scheduler.enable('JOB_N',true);
    COMMIT;
    END;
    /
    ```
    where `JOB_1… JOB_N` are the names of previously stopped jobs.

8.1.6 Database Logging

Oracle DBMS operation in Traffic Monitor is logged with a script. This script allows you to set the logging level and export log data to a CSV file. The script uses settings of the current configuration file: `/opt/iw/tm5/csw/database.conf`. You should run the script from the directory where it is stored: **/** `opt/iw/tm5/csw/oracle/scripts/sys_log/sys_log.sh`.

The script includes the following commands:

- To set the logging level: `debug, info, warning, error –` run the command: `level`
  If you do not specify the second parameter, the current logging level is displayed.
- To export the log body (V_SYS_LOG representation) to a CSV file, run the command: `export`
  If necessary, you can set the number of lines to be exported:
  `export <the number of lines>`

On how logs are stored on the disk, see *Rotation of database logs*.

Rotation of database logs

Rotation of database logs is the process of archiving, storing, and deleting obsolete data in logs. This process is required to save space and free up disk space. In Traffic Monitor, rotation applies to:

1. Database logs:
   a. XML file that contains a log of errors and important messages. It is located in the `/u01/app/oracle/diag/rdbms/iwtm/iwtm/alert` folder. The file is stored for one year. Rotation applies to each 10 Mb of the file. The file is deleted automatically.
   b. TRC and TRM files that contain detailed information about DB errors (trace files). They are located in the `/u01/app/oracle/diag/rdbms/iwtm/iwtm/trace` folder. Files are stored for 30 days and then they are automatically deleted.
   c. `alert_iwtm.log` - a database log file in a text format. It is stored in the `/u01/app/oracle/diag/rdbms/iwtm/iwtm/trace` folder. Rotation applies to each 100 Mb of the file. The file is rotated by the **logrotate** service and is deleted when the number of files exceeds 1000.
2. Logs of **listener** network service:
   a. XML file that contains logs of the **listener** service. The file is located in the `/u01/app/oracle/diag/tnslsnr/hostname/listener/alert` folder. The file is stored for one year. The file is stored for one year. Rotation applies to each 10 Mb of the file. The file is deleted automatically by the **crond** service.
   b. TRC and TRM files that contain detailed information about errors of the **listener** service. Files are stored for 30 days and then they are automatically deleted by the **crond** service.

c. `listener.log` - a log file of the **listener** service in a text format. The file is located in the `/u01/app/oracle/diag/tnslsnr/hostname/listener/alert` folder. Rotation applies to each 100 Mb of the file. The file is rotated by the **logrotate** service and is deleted when the number of files exceeds 100.

Settings of the **crond** service for deleting obsolete files are located in `/etc/cron.daily/delete_old_oracle_logs.cron.`

Settings of the **logrotate** service are located in `/etc/logrotate.d/iwtm-oracle-conf-server.`

8.2 PostgreSQL

To connect to DB server from Traffic Monitor server terminal, run the command as `root`:

```
psql postgres iwtm
```

To connect to DB from workstations running Windows, use pgAdmin program.

Connection information is located in **/opt/iw/tm5/csw/postgres/database.conf** configuration file.

This chapter contains instructions on administrating PostgreSQL:

- Tablespaces in InfoWatch Traffic Monitor database;
- Managing Daily Tablespaces;
- Load statistics of Daily Tablespaces;
- Database Backup;
- Database Server Routinely Maintenance;
- Database logging.

> **Note:**
>
> To change the DB user password, see the *Changing the Preset Password* article.

8.2.1 Tablespaces in InfoWatch Traffic Monitor database

General information

InfoWatch Traffic Monitor data are stored in DB as tablespaces (TS). There are two types of Traffic Monitor tablespaces:

The type of tablespace	Purpose
Main	Storing the settings that are needed for the analysis and processing of the objects (configuration, tags, colors, etc.). Managing the System using the Management Console (roles, user accounts, etc.)

The type of tablespace	Purpose
Daily	Storing objects intercepted within one day. Storing information on the results of the analysis and processing of the objects (analyzed object categories, terms, etc.)

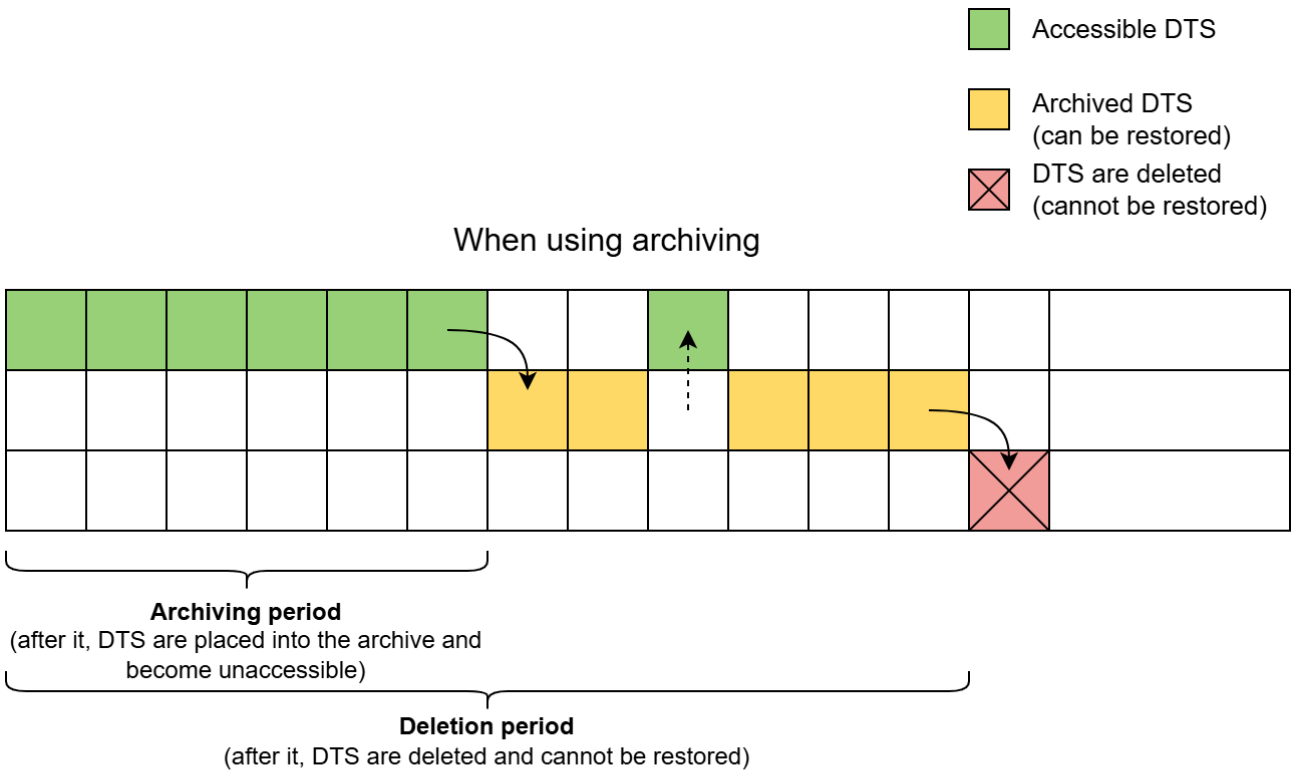
All data about the objects intercepted on a particular day (according to UTC+0) is stored in three daily tablespace (DTS) of the folowing types:

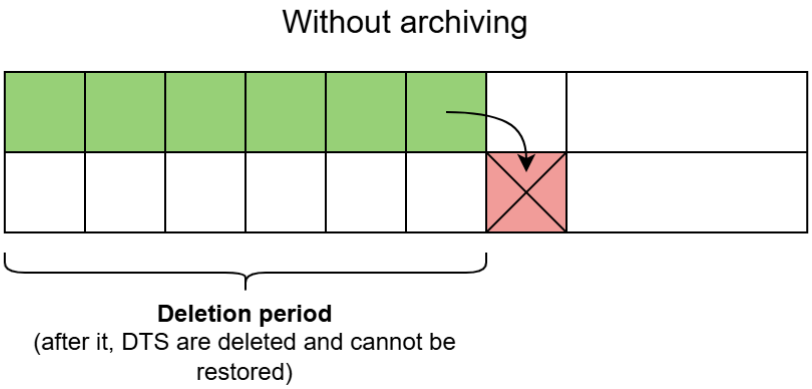
- Events without violations;
- Events with violations;
- Other (screenshots).

Daily tablespaces are created every day in such a way that the database always has DTS for the next six days, excluding the current day. All DTS automatically get the *IWTM_YYYYMMDD_X* name, where X is the DTS number (the TBS_ID value from the TBS_LIST table). The tablespace type is defined by the TBS_TYPE value: 0 – No Violations, 1-Violation, 2 - Other (screenshots).

Lifecycle of DTS can be divided into 3 stages:

1. **Ordinary storing**
   Initially, DTS are stored in the database and are in operational access. If you use Data Storage, then the event contents from this DTS such as attachments, screenshots, audio files, and so on, are stored there.
2. **Storing in an archive** (optional)
   If you need to store events for a long time, but do not need operational access to them, you can archive them before deleting.
   For this, enable automatic archiving of DTS. After the **archiving period**, they will be archived and will become unavailable. If neccesary, you can restore them (the dashed arrow on the diagram).
3. **Deleting**
   To avoid disk overflow, you can delete DTS which you do not need to store.
   For this, enable automatic deleting of DTS. After the **deletion period**, they will be deleted without the possibility of restoring.





By default, automatic archiving and deletion are disabled. You can enable them during the installation or after it. Also, you should set the DTS **archiving period** and/or the DTS **deletion period**. For details, see articles on archiving and deletion.
Configure **archiving period** and **deletion period** so that you always have enough free space on the disks, which store DTS, and they do not overflow.

If you need to archive/restore/delete a single DTS, you can do it manually in the Traffic Monitor web interface. See *InfoWatch Traffic Monitor. User Guide. Managing tablespaces*.
If the web interface is unavailable, use the **iw_space_manager** service utility for this (see *Managing Daily Tablespaces with the Console Utility*). Last operate with DTS manually in the database.

Storing

The main tablespace is stored in the `/u02/pgdata` catalogue by default.
The daily tablespaces are stored in the `/u02/pgdata1` catalogue by default.
For details on placing DTS, see *Configuring Location of Files in File System*.

On storage modes, see *Configuring Storage Modes for Tablespace Files*.

Archiving

Archiving process consists of:

- disconnecting DTS from the database;
- moving DTS to the specified catalogue (by default: `/u02/arch`).

You should place this catalogue on the separate disk to free up space on the hard drive with operational data and avoid its overflow.
When using Data Storage, binary data of the archived DTS will be moved to the Data Storage slow disk.

During archiving, data is not compressed. So, on the allocated disk, it will take about the same place. You can estimate the size of the disk for archived DTS based on how much place DTS take in operational access in their storage directories (by default: `/u02/pgdata1` in the DB and `/opt /dsdata/_seaweedfs` in Data Storage).

Archiving is described in details in the *Archiving Daily Tablespaces* section.

To access the archived DTS data, you need to restore it. See *Restoring Daily Tablespaces*.

Deleting

When automatic deletion is enabled, the System deletes DTS older than the deletion period. You specify its value separately for each DTS type: events without violations, events with violations, other (screenshots).

For details on deletion methods, see *Deleting Daily Tablespaces*.

8.2.2 Managing Daily Tablespaces

In this section:

- Configuring Location of Files in File System;
- Configuring Storage Modes for Tablespace Files;
- Archiving Daily Tablespaces;
- Restoring Daily Tablespaces;
- Deleting Daily Tablespaces;
- Managing Daily Tablespaces with the Console Utility.

Configuring Location of Files in File System

By default, the installation program (see *InfoWatch Traffic Monitor. Installation Guide* document) sets the usage of one directory for storing tablespaces: `/u02/pgdata1`.

> **Note:**
>
> For tablespace archives, the `/u02/arch` directory is used.

However, at high loads, we recommend storing daily tablespaces distributedly in different files systems (LUN, physical drives) for sequential usage. For example, if three file systems are defined, data is located as follows:

The first day – a daily tablespace is created in File system 1.

The second day – a daily tablespace is created in File system 2.

The third day – a daily tablespace is created in File system 3.

The fourth day – a daily tablespace is created in File system 1.

…

You can change the distribution of daily tablespace files (number and location) using scripts.

A sample script to change the number of separate storages for daily tablespaces:

```
begin;
select pkg_part_set_df_path_cnt('4');
commit;
```

> **Warning!**
>
> If you change the number of storages for daily tablespaces, you should also edit (add/delete) their paths.

A sample script to change paths for daily tablespaces:

```
begin;
select pkg_part_set_df_path('/test1/', 1);
select pkg_part_set_df_path('/test2/', 2);
select pkg_part_set_df_path('/test3/', 3);
select pkg_part_set_df_path('/test4/', 4);
commit;
```

> **Note:**
>
> We recommend adding “/” character at the end of the path.

A sample script to view contents of daily tablespaces:

```
select a.d, a.code, power(10, trunc(log(10, a.binary_size + a.text_size))) ||
 '-' ||
power(10, trunc(log(10, a.binary_size + a.text_size)) + 1) size_range,
count(1) cnt, sum(a.binary_size) binary_size, sum(a.text_size) text_size
from
(
select date_trunc('day', o.capture_date) d, s.display_name code, o.object_id,
coalesce(length(os.source),0)+coalesce(length(os.context),0)+
coalesce(length(o.gui_xml),0)+coalesce(length(o.preview_data),0) binary_size,
coalesce(length(o.text), 0) text_size
from object o
inner join service s on o.service_code = s.service_id and s.language = 'eng'
inner join object_source os on os.object_id = o.object_id
```

```
where o.capture_date between to_date('05.05.2014', 'dd.mm.yyyy') and
to_date('14.05.2014', 'dd.mm.yyyy')
) a
group by a.d, a.code, power(10, trunc(log(10, a.binary_size + a.text_size)))
|| '-' ||
  power(10, trunc(log(10, a.binary_size + a.text_size)) + 1)
order by 1, 2, 3;
```

> **Note:**
>
> If, along with the service information of the database, you need to retrieve the result of analyzing the size of intercepted objects in daily tablespaces, we recommend using pgAdmin.

Configuring Storage Modes for Tablespace Files

There are three modes of storing data in DB:

- **Normal** – switching to the following partition (when it is specified) happens after the previous one is overflow.
- **Fast/Slow disks** – dividing pools into fast and slow ones. Recent data are to be kept in fast pool. In a number of days specified, the System relocate them to the slow ones. Slow pool operates in Normal mode.
- **Rotate** – switching to the following partition happens every day once the previous one is overflow.

You select storage mode when installing Traffic Monitor. By default, the installer offers you **Normal** mode. You can change the storage mode later.

By default, DTS are stored in the `/u02/pgdata1` directory.

> **Note:**
>
> Modes described in this article relate to data stored in DB only. Changing mode does not affect data stored in Data Storage.

Fast/slow disks Mode

The System moves data from the fast disk to the slow one every night. If you are switching from **Normal** mode to **Fast/slow** mode, you store a large amount of DTS on your old partition, and you specify the old partition as a fast disk, then data transfer may take a long time.
If you will use the old disk as a fast one, before switching mode make sure that the following conditions are met:

- The new partition size is not less than the old one size. So, there is enough space for transferring all DTS from the old partition;
- There is free space in the old partition not less than the largest tablespace that will be transferred to the slow disk.

The System creates DTSs every day in such a way that the database always has DTS for the next six days, excluding the current day. So, if when switching from **Normal** mode to **Fast/slow** mode, you specify a new partition as a fast disk, then, for first seven days, the System will save events to the already created DTS on the slow disk. In this case, the System will create new DTSs on the fast disk.

For configuring **Fast/slow** mode, we will use the new partition `/u03/pgdata/` as a slow disk and the old partition `/u02/pgdata1/` as a fast one.

To configure "Fast/slow" storage mode (fast and slow disks):

1. Edit the `/etc/fstab` file to automatically mount the new partition to `/u03`:
   `/dev/sdd1 /u03/ ext4 defaults 1 2`
2. Create the `pgdata` folder:
   ```
   mkdir /u03/pgdata
   ```
3. Change the user:
   `chown -R postgres:postgres /u03`
4. Log into the database:
   `su - iwtm`
   `psql -p 5433 -h <DB_server_IP_address> postgres iwtm`
   `psql -p 5433 -h 127.0.0.1 postgres iwtm` – to connect to the local database
5. Specify the storage period on fast disks (7 by default):
   `select pkg_part_set_fast_days(7);`
6. Specify the path to the fast disk:
   `select pkg_part_set_fast_path('/u02/pgdata1/',1);`
7. Specify the path to the slow disk:
   `select pkg_part_set_df_path('/u03/pgdata/',1);`
8. If necessary, you can use multiple slow disks. The data will be recorded to the next slow disk when the previous one is full. To add slow disks, specify a new path:

   
   ```
   select pkg_part_set_df_path('/u03/pgdata2/',2);
   ```
9. Specify the total amount of slow disks (up to 10):
   ```
   select pkg_part_set_df_path_cnt('2');
   ```
10. Change the mode to fast-slow disks:
    `select pkg_part_set_filesys_type('fast/slow')`
    The data will be transferred from the fast disks to slow ones during a night job.
11. To start the file transfer manually, run:
    `select iwtm.pkg_part_move_fast_tablespaces_to_slow();`

> **Note:**
>
> The System allows using multiple slow disks.

> **Note:**
>
> In **Fast/slow** mode, the screenshot tablespaces are stored in slow partitions only.

To switch back to "Normal" storage mode:

1. Check the contents of the `/u02/pgdata1/` fast disk directory:
   `ls /u02/pgdata1/`
2. Check the contents of the `/u03/pgdata/` slow disk directory:
   `ls /u03/pgdata/`
3. Log into the database:
   `su - iwtm`
   `psql -p 5433 -h <DB_server_IP_address> postgres iwtm`
   `psql -p 5433 -h 127.0.0.1 postgres iwtm` – to connect to the local database
4. Change the storage period on fast disks:
   `select pkg_part_set_fast_days(-1);`
   The "-1" value is specified to transfer tablespaces for today.
5. Run the task to transfer the tablespaces from fast disks to slow disks:
   `select iwtm.pkg_part_move_fast_tablespaces_to_slow();`
6. Check the contents of the fast disks directory. The tablespaces that are created for future dates should remain. Such tablespaces should be manually deleted by using the **iw_space_manager** service:
   `iw_space_manager tbs:delete <tbs_id>`
   or via the database (if you do not use Data Storage):
   `select pkg_part_delete_tablespaces(<tbs_id>);`
7. Run the task (cron) of the **iw_space_manager** service to create DTSs imidiately:
   `iw_space_manager cron`
   or start a job in the database (if you do not use Data Storage):
   `/opt/iw/tm5/bin/dbtools/dbconf-jobs-postgres.sh start iwtm_add_parts`
8. Check that all tablespaces have been transferred from the fast disks:
   `ls /u02/pgdata1/`
9. Check that the tablespaces for future dates have been created on the slow disks:
   `ls -alt /u03/pgdata/`
10. Enable **Normal** mode:
    `select pkg_part_set_filesys_type('normal');`

Rotate Mode

To configure "Rotate" storage mode (daily rotation):

1. Mount the partition and make sure that it can be mounted automatically on reboot.
2. Create the `pgdata` folder:
   ```
   mkdir /u03/pgdata
   ```
3. Change the user:
   `chown -R postgres:postgres /u03`
4. Log into the database:
   `su - iwtm`
   `psql -p 5433 -h <DB_server_IP_address> postgres iwtm`
   `psql -p 5433 -h 127.0.0.1 postgres iwtm` – to connect local database

5. Add a path to the second partition:
   `select pkg_part_set_df_path('/u03/pgdata/',2);`
6. Specify a new number of paths:
   `select pkg_part_set_df_path_cnt(2);`
7. Switch to **rotate** mode:
   `select pkg_part_set_filesys_type('rotate');`

```
          postgres=# select * from setting where setting like 'df_%';
setting | value | editable
------------+---------------+----------
df_path1 | /u02/pgdata1/ | 1
df_path2 | /u03/pgdata/ | 1
df_path_cnt | 2 | 1
df_filesys_type | rotate | 1
```

> **Note:**
>
> Parameters of modes are stored in the `setting` table of the database. Their specifics are described in the *Tablespaces data storage settings. Data storage partition* Knowledge Base article.

Archiving Daily Tablespaces

Archiving process consists of disconnecting DTS from the database and moving them to the specified catalogue (by default: `/u02/arch`). You should place this catalogue on the separate disk to free up space on the hard drive with operational data and avoid its overflow. You can restore this data if necessary.

In the System, there are following ways of archiving DTS:

- Automatically after the specified period is expired (see *Automatic Archiving of Daily Tablespaces*);
- Manual archiving of single DTS:
  - In the Traffic Monitor web interface. See *InfoWatch Traffic Monitor. User Guide. Managing tablespaces*;
  - Using the console utility of the **iw_space_manager** service (see *Managing Daily Tablespaces with the Console Utility*);
  - In the database (see *Manual Archiving of Daily Tablespaces in the Database*).

If you need to archive a single DTS, we recommend to do it in the Traffic Monitor web interface. If the web interface is unavailable, use the **iw_space_manager** service utility. Last use manual archiving of DTS in the database.

Archiving when using Data Storage

If you use Data Storage, then, when archiving DTS, binary data will be moved to the Data Storage slow disk.
If the slow disk is not connected, then, in Traffic Monitor web interface, archived daily tablespace will have the "*Archived*" status. But in this case, binary data of DTS will be stored on the fast disk of the Data Storage server,

inside the directory for placing binary data. You specify this directory when installing the storage (by default: `/opt/dsdata/_seaweedfs`).

You specify the path to the slow disk when installing Data Storage. For details, see *InfoWatch Traffic Monitor. Installation Guide. Installing Platform*.
To add the path to the Data Storage slow disk after installing the storage:

1. Create a directory and mount the slow disk to it:

```
mkdir -p /mnt/slowdisk
mount /dev/<your_slowdisk> /mnt/slowdisk
```

2. Set a user and access permissions for the directory, where you have mounted the slow disk:

```
chown 1000 /mnt/slowdisk
chmod 700 /mnt/slowdisk
```

3. Open the Data Storage deployment for editing. For this use the command:

```
kubectl -n infowatch edit deployment datastorage-<node_label>
```

   where `<node_label>` (hereafter) is the label of the node where the storage is installed. For the central node or, if the storage is installed not in the cluster of Platform products, use the label `central`. For the office node, the label is specified during installation (`office` by default).
4. Add the following lines to the `containers` -> `volumeMounts` section:

```
      - mountPath: /var/lib/dataslow
        name: datastorage-dataslow-host-volume-<node_label>
```

5. Add the following lines to the `volumes` section:

```
      - hostPath:
          path: /mnt/slowdisk
          type: ""
        name: datastorage-dataslow-host-volume-<node_label>
```

6. Save the changes and exit the editor.

Automatic Archiving of Daily Tablespaces

By default, automatic DTS archiving is disabled. To make the System archive DTS according to schedule, enable this function and set the time after which DTS must be archived. Details are described further.
Every day, at 00:00 server time, the **iw_space_manager** service procedure will archive DTS that satisfy the following condition:

The date of a tablespace is less or equal to the difference between the current date and the specified time interval for archiving the tablespace.

So, all DTS older than the specified value will be archived.

Warning!

If months or years are used as the time interval, we recommend using the maximum value of the interval in calculations. For a month – 31 days. For a year – 366 days. For example, to archive DTS older than 4 years, specify the interval 366 * 4 = 1464 days.

Before you enable automatic archiving, check that the archive directory is specified correctly:

```
select value
from setting
where setting = 'archive_path';
```

The `postgres` user must be the owner of the directory.

If the directory is specified incorrectly, then set it properly with the following command:

```
begin;
select sp_setting_set('archive_path', '/test/archive/');
commit;
```

To enable automatic archiving of tablespaces (by default, the option is disabled), you can use the following scripts (you should run scripts on behalf of the owner of the database schema):

- For the daily tablespace that stores objects with *Violation* status:

  ```
  begin;
  select sp_setting_set('violation_archive_enabled', '1');
  select sp_setting_set('violation_archive_period', 'D');
  commit;
  ```

- For the daily tablespace that stores objects with *No Violation* status:

  ```
  begin;
  select sp_setting_set('noviolation_archive_enabled', '1');
  select sp_setting_set('noviolation_archive_period', 'D');
  commit;
  ```

- For the daily tablespace that stores *screenshots*:

  ```
  begin;
  select sp_setting_set('other_archive_enabled', '1');
  select sp_setting_set('other_archive_period', 'D');
  commit;
  ```

  where:
  1 indicates that automatical archiving is enabled (to disable it, specify 0 instead);
  D is a number of days after which the daily tablespace is archived by the System. This number should be smaller than the set number of days before deleting the daily tablespace (see *Deleting Daily Tablespaces*).

Manual Archiving of Daily Tablespaces in the Database

It is recommended to archive single DTS via web interface or using the **iw_space_manager** utility.

The order of archiving daily tablespaces manually in the database is described below.

> **Important!**
>
> In this method of archiving, data stored in the Data Storage are not archived.

Selecting daily tablespaces and data files

You can get a list of tablespaces that can be disconnected by running queries to the database on behalf of the database schema owner. The queries are made according to the archiving policy of your company.

Example. To select daily tablespaces with basic attributes (name, identifier, path, types, creation date, size), you can use the following query:

```
SELECT t.tbs_name, t.tbs_id, t.tbs_type, t.part_date, pg_tablespace_location(oid),
pg_size_pretty(pg_tablespace_size(spcname)), t.status, t.note
FROM  pg_tablespace, iwtm.tbs_list t
WHERE t.tbs_name = pg_tablespace.spcname AND t.status IN (0, 3);
```

```
tbs_name  | tbs_id | tbs_type | part_date  | pg_tablespace_location | pg_size_pretty | status |       note
----------+--------+----------+------------+------------------------+----------------+--------
+--------------------
   iwtm_2 |      2 |        1 | 2017-03-30 |     /u02/pgdata/iwtm_2 |         692 kB |      0 | Partitions
created
   iwtm_3 |      3 |        2 | 2017-03-30 |     /u02/pgdata/iwtm_3 |         692 kB |      0 | Partitions
created
   iwtm_4 |      4 |        0 | 2017-03-31 |     /u02/pgdata/iwtm_4 |          25 MB |      0 | Partitions
created
   iwtm_5 |      5 |        1 | 2017-03-31 |     /u02/pgdata/iwtm_5 |         692 kB |      0 | Partitions
created
   iwtm_6 |      6 |        2 | 2017-03-31 |     /u02/pgdata/iwtm_6 |         692 kB |      0 | Partitions
created
   iwtm_7 |      7 |        0 | 2017-04-01 |     /u02/pgdata/iwtm_7 |         692 kB |      0 | Partitions
created
   iwtm_8 |      8 |        1 | 2017-04-01 |     /u02/pgdata/iwtm_8 |         692 kB |      0 | Partitions
created
   iwtm_9 |      9 |        2 | 2017-04-01 |     /u02/pgdata/iwtm_9 |         692 kB |      0 | Partitions
created
```

> **Warning!**
>
> To get the list of files, we highly recommend using the sample query described in the article. If you use a different query, the list of files may be incomplete.

Disconnecting Daily Tablespaces from the Database

Warning!

You should not disconnect daily tablespaces when the iwtm_iwtm_add_parts, iwtm_iwtm_delete_tablespaces, iwtm_iwtm_archive_tablespaces jobs are running as it may cause data corruption.

1. Call the procedure on behalf of the owner of the database schema:
   `select pkg_part_archive_tablespace(N);`
   where `N` is a value of `tbs_id` attribute of the required tablespace from the `tbs_list` table.
   After running this script, the daily tablespace status will change to *`Offline`*.
2. Repeat the procedure call for each daily tablespace you want to disconnect.

Moving Data Files

Move catalogs with archived tablespaces which belong disconnected daily tablespaces to a different storage.

To get the list of disconnected tablespaces, use the following command:
`select * from tbs_list where status = 10;`
Data files are located in the directory:
`select value from setting where setting = 'archive_path';`

Restoring Daily Tablespaces

Warning!

A tablespace can only be restored in the database schema, from which it was disconnected (even if this schema was updated). The restoration of the tablespace after total database schema reinstallation is impossible.

In the System, there are the following ways of restoring DTS:

- In the Traffic Monitor web interface. See *InfoWatch Traffic Monitor. User Guide. Managing tablespaces*;
- Using the console utility of the **iw_space_manager** service. See *Managing Daily Tablespaces with the Console Utility*;
- In the database – this method is described below. It is recommended to use this method last.

Moving data files

To restore daily TS, you must move data files of this tablespace from a data carrier to the folder which path you can obtain using the following command:

```
select value from setting where setting = 'archive_path'
```

Warning!

Make sure that the *postgres* user has read and write privileges for the directory to move the data files to.

Connecting the daily tablespace

Warning!

Do not connect daily TS during iwtm_iwtm_add_parts, iwtm_iwtm_delete_tablespaces and iwtm_iwtm_archive_tablespaces jobs. This may cause data loss.

1. On behalf of the owner of the database schema, call the procedure:

   ```
   select pkg_part_restore_tablespace(N);
   ```

   where `N` is ID of the tablespace (specified in tbs_list).
   After completing the procedure, the status of TS will change to *Restored*.
2. Repeat the procedure call for each daily TS you want to enable.

Deleting Daily Tablespaces

In the System, there are the following ways of deleting DTS:

- Automatic, according to schedule. Described below.
- Manual deleting of single DTS:
  - In the Traffic Monitor web interface. See *InfoWatch Traffic Monitor. User Guide. Managing tablespaces*;
  - Using the console utility of the **iw_space_manager** service. See *Managing Daily Tablespaces with the Console Utility*;
  - In the database. Described below.

We recommend using scheduled deletion. Otherwise, disk overflow can occur or the big amount of data for deletion can build up. It will result in increasing the time for running this procedure in future and consequently increasing the idle time of the Traffic Monitor Server.

Warning!

Deleting tablespaces cannot be undone. If you complete the operation, you will not be able to restore the deleted data.
The System deletes both DTS data from the storage directory (by default, `/u02/oradata1`, or `/u02/arch` for the archived ones) and information on DTS from the database. If you use Data Storage, data from it will be deleted as well.

While deleting tablespaces, access to the corresponding tables is closed. For this reason, log files of the **iw_deliver**, **iw_x2db**, **iw_updater** processes may contain database access errors. Once tablespaces are deleted, these processes automatically restore access to the database.

Automatic deletion

By default, automatic DTS deletion is disabled. To make the System delete DTS according to schedule, enable this function and set the time after which DTS must be deleted. Details are described further.
Every day, at 00:00 server time, the **iw_space_manager** service procedure will delete DTS that satisfy the following condition:

The date of a tablespace is less or equal to the difference between the current date and the specified time interval for deleting the tablespace.

So, all DTS older than the specified value will be deleted.

> **Warning!**
>
> If months or years are used as a time interval, then we recommend you to apply the maximal interval value in the counting: 31 days for a month and 366 days for a year. For example, to delete DTS older than 4 years, specify 366*4=1464 days interval.

Enabling automatic deletion

Automatic deletion according to schedule is configured separately for each DTS type.
To enable it, set the deletion `delete_enabled` flag and specify the period of storing DTS `delete_period`.
For this, run the following scripts on behalf of the database scheme owner:

- For DTS storing objects with the *Violation* status:

```
begin;
select sp_setting_set('violation_delete_enabled', '1');
select sp_setting_set('violation_delete_period', 'D');
commit;
```

- For daily TS storing objects with the *No Violation* status:

```
begin;
select sp_setting_set('noviolation_delete_enabled', '1');
select sp_setting_set('noviolation_delete_period', 'D');
commit;
```

- For daily TS storing *screenshots*:

```
begin;
select sp_setting_set('other_delete_enabled', '1');
select sp_setting_set('other_delete_period', 'D');
commit;
```

where:
1 indicates that automatic deletion is enabled (to disable, specify 0 instead of 1);
D is a number of days at the end of which the DTS will be deleted by the System. This number should be larger than the set number of days before archiving daily tablespaces (see *Automatic Archiving of Daily Tablespaces*).

Deleting daily tablespaces with a script

As well, you can configure automatic deletion of daily tablespaces, use the `/opt/iw/tm5/bin/dbtools/dbconf-iwdrop-postgres.sh` or `/opt/iw/tm5/bin/dbtools/dbconf-jobs-postgres.sh` script. You can use the following commands:

Purpose	Command
Set up deletion period	`/opt/iw/tm5/bin/dbtools/dbconf-iwdrop-postgres.sh set violation|noviolation|other <days> [-v]`
Enable automatic deletion	`/opt/iw/tm5/bin/dbtools/dbconf-iwdrop-postgres.sh enable violation|noviolation|other [-v]`
Disable automatic deletion	`/opt/iw/tm5/bin/dbtools/dbconf-iwdrop-postgres.sh disable violation|noviolation|other [-v]`
View the state of settings	`/opt/iw/tm5/bin/dbtools/dbconf-iwdrop-postgres.sh show [-v]`
Delete all DTS	`/opt/iw/tm5/bin/dbtools/dbconf-jobs-postgres.sh start iwtm_delete_tablespace`

Example:

To enable automatic deletion of the DTS, that store objects with the *"No violation"* status and are older than 15 days, run the commands:

```
/opt/iw/tm5/bin/dbtools/dbconf-iwdrop-postgres.sh set noviolation 15 [-v]
/opt/iw/tm5/bin/dbtools/dbconf-iwdrop-postgres.sh enable noviolation [-v]
```

Manual deletion in the database

If you need to immediately launch the procedure of automatic DTS deletion (e.g., due to file system space shortage the interval of deletion was changed, but the next launch of the job will not happen soon), run the following script on behalf of the database scheme owner:

```
select pkg_part.delete_tablespaces();
```

As a result, the System will delete DTS for which scheduled deletion is enabled and the period of storing is specified.

The command above affects data in DB only. If you use Data Storage, launch the procedure of automatic DTS deletion using the **iw_space_manager** service:

```
iw_space_manager cron
```

To delete one DTS by its identifier (ID), run the following script:

```
select pkg_part.delete_tablespace(<ID>);
```

This command affects data in DB only. If you use Data Storage, delete DTS using the **iw_space_manager** service:

```
iw_space_manager tbs:delete <tbs_id>
```

Managing Daily Tablespaces with the Console Utility

To manage daily tablespaces via the command line, use the console utility that calls the **iw_space_manager** service. The service is available for the **iwtm** user and is enabled by default.

When using Data Storage, the **iw_space_manager** service archives/restores/deletes binary data from the storage as well.

Commands and parameters of the service:

Command	Description
`tbs:archive [options] [--] <id>`	Starts archiving a daily tablespace with its ID specified. An example on how to archive a daily tablespace with ID=123: `iw_space_manager tbs:archive 123`
`tbs:create <type> <capture>`	Creates a daily tablespace with the following data specified: • Type: 0 – no violation, 1 – violation, 2 – screenshots • Creation date in the YYYY-MM-DD format An example on how to create a new daily tablespace with violations on September 1, 2023: `iw_space_manager tbs:create 1 2023-09-01`
`tbs:delete [options] [--] <id>`	Deletes a daily tablespace with its ID specified
`tbs:get [options]`	Gets the list of existing daily tablespaces. Additional parameters: • `--limit=LIMIT` – sets the maximal number of items to display on the screen • `--offset=OFFSET` – hides the first N items (N=0 by default) • `--order=ORDER` – sorts the daily tablespace list by its name and creation time: ASC – ascending, DESC – descending An example on how to get a list of 20 daily tablespaces sorted by name and descending date: `iw_space_manager tbs:get -- limit=20 --order="TBS_NAME:ASC,PART_DATE:DESC"`

Command	Description
`tbs:restore [options] [--] <id>`	Restores an archived daily tablespace with its ID specified. An example on how to restore a daily tablespace with ID=123 from an archive: `iw_space_manager tbs:restore 123`
	With all the commands listed above, you can use the following common parameters: • `-f, --force` – runs the command without the validation check in case of a database emergency. Enables you to perform operations on tablespaces where date of event interception coincides with today's date; • `--used-configs` – runs the command with the parameters from the configuration file; • `-p, --configpath [=CONFIGPATH]` – displays the configuration file path. By default, it is `opt/iw/tm5/etc` ; • `--force-config-load` – force loads the configuration file even if the query parameters are specified.
`tbs:clearLostBuckets`	Deleting obsolete, lost or unused buckets of table spaces (available starting from Traffic Monitor 7.10.1)

To get logs and debug messages, run the service with the parameters below:

- `-v` – displays usual logs;
- `-vv` – displays logs with extended detalization;
- `-vvv` – displays logs with full detalization.

To get a list of available commands, run the command:

`iw_space_manager list`

8.2.3 Load statistics of Daily Tablespaces

To analyze a load of daily tablespaces, you can use the `/opt/iw/tm5/bin/dbtools/tablespace_stat_postgres/tablespace_stat_postgres.sh` script. With its help, you can get the statistics on TBS content for the selected period. The script executes one of the six scenarios and in each case it collects different information. The scenarios are described in more detail *below*.

To display the information on the script usage:

1. Go to the directory where the script is located by running the command:
   `cd /opt/iw/tm5/bin/dbtools/tablespace_stat_postgres/`
2. Run the script without parameters by running the command:
   `./tablespace_stat_postgres.sh`

As a result, you will see the help information:

```
tablespace_stat.sh: usage: [ report_number ] [ -e ]

report_number:
1 - Period stats
2 - Period stats with count of objects
3 - Statistics on a given tbs id by event type
4 - Period stats by event type
5 - Period stats by person
6 - Statistics on a given person id

-e, --export : for export to csv
```

To use the script:

1. Go to the directory where the script is located by running the command:
   `cd /opt/iw/tm5/bin/dbtools/tablespace_stat_postgres/`
2. Run the script by running the command:
   `./tablespace_stat_postgres.sh <scenario_number> <-e>`
   where the scenario number is a digit from 1 to 6;
   `-e` or `--export` is a key that you can specify to unload the results of script execution in .csv format.
   Example of a command running the script:
   `./tablespace_stat_postgres.sh 1`
3. Specify the parameters depending on a scenario:
   - For scenarios 1, 2, 4, 5, 6 – a period for which the statistics should be displayed:
     a. Specify the lower bound of the period in the dd.mm.yyyy format and press **Enter**;
     b. Specify the upper bound and press **Enter**;

     > **Note:**
     >
     > If you do not specify values for bounds and just press **Enter** twice then, by default, statistics for last two weeks will be displayed.
     > If you specify only the lower bound then information for the period from the specified date to the current one will be displayed.

   - For scenario 3 – ID of the tablespace;
   - For scenario 6 – person's ID.

In case of unloading the results of script performing, the file is saved in the directory where the script is located. The file has the following name:
`report_<scenario_name>_<date>.csv`
where `<scenario_name>` has a value depending on the scenario number:

Scenario number	Scenario name	Short description
1	tbs_stat_period	Statistics for the period of time
2	tbs_stat_period_count	Statistics for the period of time including count of events in TBS
3	tbs_stat_by_tbs_id	Statistics on a particular TBS
4	tbs_stat_by_event_type	Statistics on event types for the period

5	tbs_stat_by_persons	Statistics on top-10 persons
6	tbs_stat_by_person_id	Statistics on a particular person

`<date>` - date of script execution. It has the `<yyyy-mm-dd_HH-mm-ss>` format.
Example of the unloading file name: `report_tbs_stat_period_count_2024-12-17_18-20-39.csv`

Scenarios of script execution

Scenario 1

As a result of scenario 1 execution, statistics on space that tablespaces take up in DB for the period of time will be displayed.

```
Period tablespace stats

Date format: dd.mm.yyyy (empty - no filter. Default - 14 days)
Enter value for start_date: 18.12.2024
Enter value for end_date: 19.12.2024

 tbs_type | part_date  | tbs_size | avg_tbs_size
----------+------------+----------+--------------
        2 | 18.12.2024 | 804 kB   | 897 kB
        1 | 18.12.2024 | 804 kB   | 897 kB
        0 | 18.12.2024 | 1364 kB  | 897 kB
        2 | 19.12.2024 | 804 kB   | 897 kB
        1 | 19.12.2024 | 804 kB   | 897 kB
        0 | 19.12.2024 | 804 kB   | 897 kB
(6 rows)
```

The following information will be shown in columns:

tbs_type	TBS type: 0 - No Violations; 1 - Violation; 2 - Other (screenshots).
part_date	Date of TBS creation
tbs_size	TBS size
avg_tbs_size	Average TBS size for the specified period of time

Scenario 2

As a result of scenario 2 execution, more detailed statistics on tablespaces for the period of time will be displayed.

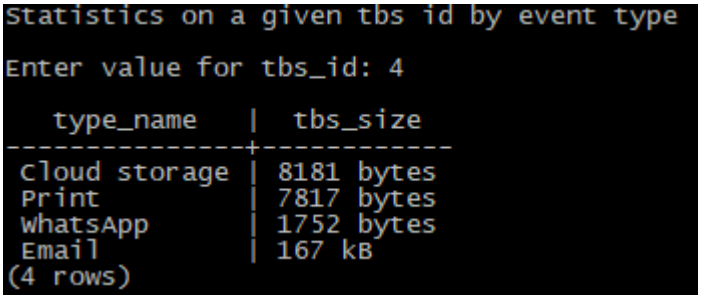
```
Period tablespace stats with count of objects

Date format: dd.mm.yyyy (empty - no filter. Default - 14 days)
Enter value for start_date: 18.12.2024
Enter value for end_date: 19.12.2024

 tbs_id | part_date  | tbs_type | tbs_status | count_objects | tbs_size
--------+------------+----------+------------+---------------+----------
      6 | 18.12.2024 |        2 | 0 - online |             0 | 804 kB
      5 | 18.12.2024 |        1 | 0 - online |             0 | 804 kB
      4 | 18.12.2024 |        0 | 0 - online |             8 | 1364 kB
      9 | 19.12.2024 |        2 | 0 - online |             0 | 804 kB
      8 | 19.12.2024 |        1 | 0 - online |             0 | 804 kB
      7 | 19.12.2024 |        0 | 0 - online |             0 | 804 kB
(6 rows)
```

The following information will be shown in columns:

tbs_id	TBS ID
part_date	Date of TBS creation
tbs_type	TBS type: 0 - No Violations; 1 - Violation; 2 - Other (screenshots).
tbs_status	TBS status: 0 - online (connected, in normal state); 3 - read only (only for reading); 5 - before exchange (before disconnecting for archiving); 7 - prepared offline (prepared to be disconnected for archiving); 9 - disconnected (disconnected before moving to archive); 10 - offline (archived and disconnected); 11 - plugged (plugged in before restoration); 13 - before update (before updating for restoration); 15 - after update (after updating for restoration); 17 - before revert (before restoring from archive); 20 - restored (restored from archive).
count_objects	Count of events in a TBS
tbs_size	TBS size

Scenario 3

As a result of scenario 3 execution, statistics on a particular TBS will be displayed.

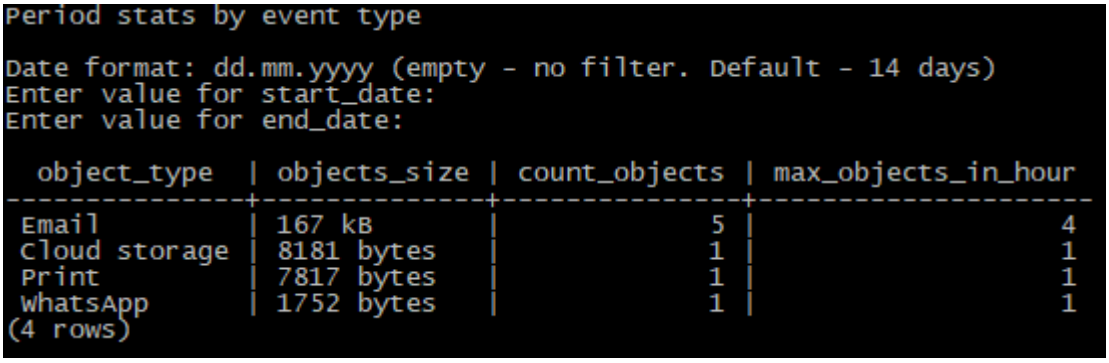


The following information will be shown in columns:

type_name	Event type
tbs_size	Overall size of the events of the particular type in TBS

Scenario 4

As a result of scenario 4 execution, statistics on event types for the specified period of time will be displayed.



The following information will be shown in columns:

object_type	Event type
objects_size	Overall size of the events of the particular type
count_objects	Count of the events of the particular type
max_objects_in_hour	Maximum count of events per hour during the day

Scenario 5

As a result of scenario 5 execution, statistics on top-10 persons for the specified period of time will be displayed. Events with participation of top-10 persons take up the largest space in DB.

```
Period stats by persons

Date format: dd.mm.yyyy (empty - no filter. Default - 14 days)
Enter value for start_date:
Enter value for end_date:

               person_id                |  display_name  | object_count | objects_size
----------------------------------------+----------------+--------------+--------------
 B37174C262444412B7AC8ECDCA544BD200000000 | Ivanov Ivan    |            3 | 109 kB
 099914C8CFD6416CB734C7721A2CF37600000000 | Petrov Petr    |            2 | 62 kB
 554BEE71D78B49CE80DA30301022AC9400000000 | Margaret Smith |            1 | 47 kB
(3 rows)
```

The following information will be shown in columns:

person_id	Person ID
display_name	Person display name
object_count	Count of events
objects_size	Overall size of the events with participation of the person

Scenario 6

As a result of scenario 6 execution, statistics on particular person for the specified period of time will be displayed.

```
Statistics on a given person id

Enter value for person_id: B37174C262444412B7AC8ECDCA544BD200000000
Date format: dd.mm.yyyy (empty - no filter. Default - 14 days)
Enter value for start_date:
Enter value for end_date:

'Ivanov Ivan'
 object_type | object_count | objects_size
-------------+--------------+--------------
 Email       |            3 | 109 kB
(1 row)
```

The following information will be shown in columns:

object_type	Event type
object_count	Count of events
objects_size	Overall size of the events with participation of the person

8.2.4 Database Backup

To minimize the risk of data loss, we recommend to backup the database on a monthly basis. To store backups, we recommend you to use dedicated storage system.

For non-database administrators who do not have standard backuping methods, we recommend creating a “cold backup” at the stopped database. You may also use this procedure to migrate the database from one server to another.

If, in addition to database, you use Data Storage, i.e. a storage of binary data, then back up the storage together with DB. This procedure is decribed together with DB backup creation.

This section further covers:

- Creating Database Backup;
- Database Backup.

Creating Database Backup

Creating database backup (cold backup procedure) is performed in the following order:

- Defining the Size of Backup;
- Checking Backup Storage;
- Creating Backup Directories;
- Stopping IWTM System;
- Stopping PostgreSQL;
- Copying Database Files to Backup Storage;
- Creating Data Storage Backup.

Defining the Size of Backup

To define size of the future archive, it is necessary to find out summary size of following catalogues:

- a catalogue with PostgreSQL DB;
- a catalogue with the main tablespace;
- catalogues with daily tablespaces catalogues;
- the `/opt/dsdata/_seaweedfs` catalogue for storing data of Data Storage if you use it.

For this:

1. On the DB server, enter the system as the **root** user.
2. Get summary catalogue size:
   ```
   du -sx -BM /u01/postgres/ /u02/pgdata /u02/pgdata1 /u02/arch
   ```
3. On the Data Storage server, define size of the `/opt/dsdata/_seaweedfs` catalogue similarly.

If you use another storage mode, the data can be stored in different directories (for details, see *Configuring Storage Modes for Tablespace Files*).

Checking Backup Storage

To check if the selected backup storage is suitable, take the following steps:

1. Mount the external storage to the DB server. Make sure that the following requirements are met:
   - The external storage, the database and Data Storage are located on different computers.
   - Computers with servers and databases to be backed up can access the external storage.
   - The external storage has more free space on the hard disk than the backup size.
2. Define the amount of free space on the hard disk:
   a. Run the following command as **root**:
      ```
      OS>df -h
      ```
   b. Make sure that the mounted section has more free space than the backup size.

Creating Backup Directories

> **Warning!**
>
> Backup directories should be located on a computer other than the database or Data Storage is located.

1. Log the system as **root** user.
2. Create the directory for backup files storage:
   `mkdir /opt/IWTM_Backup_Files`
3. Create subdirectories:
   ```
   mkdir /opt/IWTM_Backup_Files/postgres
   mkdir /opt/IWTM_Backup_Files/pgdata
   mkdir /opt/IWTM_Backup_Files/pgdata1
   mkdir /opt/IWTM_Backup_Files/arch
   ```

Stopping IWTM System

Stopping IWTM system is not required but we recommend doing it especially if there is little free space on IWTM server.

To stop IWTM system:

1. On each computer that hosts IWTM services, log in as **root**.
2. Stop all running Traffic Monitor Server services:
   `iwtm stop`
   You can start services later (when backup is finished) using the following command:
   `iwtm start`

Stopping PostgreSQL

> **Important!**
>
> Before backuping the database files, you should obligatorily stop the PostgreSQL database.

To stop PostgreSQL database:

1. On the computer that hosts the database, log on as the **root** user.
2. In the command line, enter:
   ```
   systemctl stop pgagent-15
   systemctl stop postgresql-15
   ```
   You can start services later (once backup is finished), using the following command:
   ```
   systemctl start pgagent-15
   systemctl start postgresql-15
   ```

Copying Database Files to Backup Storage

To copy database files:

1. Stop all PostgreSQL services (*Stopping PostgreSQL*). To make sure that all services are stopped, run the following command:

`ps aux | grep postgre`
If PostgreSQL services are not stopped, backup files can be damaged and unsuitable for recovery.
2. On the computer that hosts the database, copy data from DB directories to previously created directories. You need to copy all the contents of the directories including links.
If the system has been installed with the installer and default settings are not modified, then:
   - copy contents of the **/u01/postgres/** directory to the **/opt/IWTM_Backup_Files/postgres** directory
   - copy contents of the **/u02/pgdata/** directory to the **/opt/IWTM_Backup_Files/pgdata** directory
   - copy contents of the **/u02/pgdata1/** directory to the **/opt/IWTM_Backup_Files/pgdata1** directory
   - copy contents of the **/u02/arch/** directory to the **/opt/IWTM_Backup_Files/arch** directory

Important!

Use only the following devices as file storages:

- an external drive with Ext4 or XFS file system
- a remote block device connected via iSCSI or NFS protocol
- a locally connected block device with Ext4 or XFS file system

Creating Data Storage Backup

In Data Storage, Traffic Monitor data are stored in buckets. To back up the storage of binary data, you need to back up these buckets. This procedure consists of the following steps:

1. Enabling functionality of backup
2. Getting a token to work with the Data Storage service
3. Getting a list of buckets for backup
4. Creating the backup task
5. Checking successful completion of backup

During the backuping process you need to send requests using the **curl** utility. When doing this, you can specify the following values as a server address `<ADDRESS>`:

- **localhost:40731**, if you send requests from the server where Data Storage is installed.
- **<server_IP_address>:40732**, if you send requests from another server.

Enabling functionality of backup

To enable creating a backup of Data Storage, you need to specify a directory for that.

We recommend to do it when you install the storage. For this, run the installer using the `--dsbackupspath` parameter with the directory for backups.
Example: `./setup.py install --dsbackupspath=/opt/dsbackups`

To specify the directory after installation, add it to the Data Storage deployment. For this:

1. On the Data Storage server, create a directory for storing a backup:

```
mkdir /opt/dsbackups
```

2. Mount an external storage for backups into this directory. Requirements for the backup storage are described in the *Checking Backup Storage* article.
3. Set a user and access permissions for the created directory:

```
chown 1000 /opt/dsbackups
chmod 700 /opt/dsbackups
```

4. Open the Data Storage deployment for editing. For this use the command:

```
kubectl -n infowatch edit deployment datastorage-<node_label>
```

   where `<node_label>` (hereafter) is the label of the node where the storage is installed. For the central node or, if the storage is installed not in the cluster of Platform products, use the label `central`. For the office node, the label is specified during installation (`office` by default). The following actions depends on a text editor that you use. In our example, we use **vi**. To enter insert mode, press **i**.
5. Add the following lines to the `containers` -> `volumeMounts` section:

```
- name: datastorage-databackups-host-volume-<node_label>
  mountPath: /var/lib/databackups
  mountPropagation: HostToContainer
```

   In this and the next step, indent with spaces and do not use a tab.
6. Add the following lines to the `volumes` section:

```
- name: datastorage-databackups-host-volume-<node_label>
  hostPath:
    path: /opt/dsbackups
```

7. Save the changes and exit the editor.
   For example, if you use the **vi** editor, press **Esc** and type `:wq`.

Getting a token to work with the Data Storage service

To work with the Data Storage service, you need to get a Bearer token for HTTP authorization. In requests to Data Storage, specify it in the Authorization header.
The token value can become outdated. So in response to a request, the service returns the 440 HTTP code. In this case, get the token again.

To get the token, send the following request to Data Storage:

```
curl -X GET -I --silent "http://<ADDRESS>/v2/logon?extAPIToken=<DS_user_token>" 2>&1 | grep
'^Authorization'
```

where `<ADDRESS>` is the server address. For details, see above.
`<DS_user_token>` is the Data Storage authorization token. The default value: `1exwo92iwpczl1hy6jxy`.
To learn more, see *Configuring Connection with Data Storage*.

As a result, you will see the Authorization header in the following format:
`Authorization: Bearer <token_value>`

Getting a list of buckets for backup

To get a list of identifiers of Data Storage buckets which store Traffic Monitor data, connect to the Traffic Monitor database and run the query:

```
select distinct(storage_bucket_id) from iwtm.tbs_list;
```

If you need to know a list of all existing Data Storage buckets (their bucketID identifiers), send the following request to the storage service:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets?filter=*" 2>&1
```

where `<AUTH_HEADER>` is the Authorization header received on the previous step;
`<ADDRESS>` is the server address. For details, see above.

Results in the JSON format may look as follows:


```
{
  "data": [
    {
      "backendType": "seaweedfs",
      "id": "barbucket",
      "isDumped": false,
      "isInaccurate": false,
      "isSlowDrive": false,
      "size": 27632
    },
    {
      "backendType": "seaweedfs",
      "id": "bazbucket",
      "isDumped": false,
      "isInaccurate": false,
      "isSlowDrive": false,
      "size": 27633
    },
    {
      "backendType": "file",
      "id": "oldbucket",
      "isDumped": false,
      "isInaccurate": false,
      "isSlowDrive": false,
      "size": 27634
    }
  ]
}
```


Also you can filter the bucket list with a regular expression using the **jq** utility. For example, the following request will find all buckets whose bucket ID value begins with the "bar" substring:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets?filter=*" 2>&1 | jq --raw-output
'.data.[] | select(.backendType=="seaweedfs") | select(.id|test("^bar")).id'
```

Creating the backup task

To create a task of bucket backup, send the following request:

```
curl -X POST --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/backups?bucketID=<BUCKET_ID>"
```

where `<AUTH_HEADER>` is the Authorization header;
`<ADDRESS>` is the server address. For details, see above;
`<BUCKET_ID>` is the bucket identifier.

In response, the ID of the created task will be displayed:

```
{
  "taskID": "fe8163f2-a0fb-4990-8bb7-87aee5c425bc"
}
```

Also you can create a bucket backup task and get the ID of this task using the **jq** utility:

```
curl -X POST --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/backups?bucketID=<BUCKET_ID>" |
jq --raw-output '.taskID'
```

Next, you can check successful completion of bucket backup using the task ID.

Checking successful completion of backup

To make sure that backuping completed successfully, check the status of the task. For this, send the request:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/backups?taskID=<TASK_ID>"
```

where `<AUTH_HEADER>` is the Authorization header;
`<ADDRESS>` is the server address. For details, see above;
`<TASK_ID>` is the task identifier.

If succesfull, you will see names of data and index files created in the backup directory:

```
{
  "bucketID": "foobucket",
  "dataFilename": "foobucket_0.tar.gz",
  "indexFilename": "foobucket_extindex.tar.gz",
  "taskStatus": "Success"
}
```

In case of failure, the error description will be displayed:

```
{
  "bucketID": "foobucket",
  "error": "permission denied: cannot extract data tarball",
  "taskStatus": "Success"
}
```

Make sure that the task is completed and has one of the following statuses: Success, Canceled, Failed.

Also you can get the task status using the **jq** utility:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/backups?taskID=<TASK_ID>" | jq
--raw-output '.taskStatus'
```

Restoring Database from Backup

To restore the database and/or Data Storage:

1. Choose the appropriate method of database recovery if necessary:
    - If the database was damaged due to the system failure or user error, restore the old database. For example, if an important file has been occasionally deleted, you can restore the database to the state when this file still existed. For details, see *Restoring to the Same Database*.
    - If the old database cannot be used any more, create a new database and recover data to it. See *Restoring to a New Database*.
2. Follow one of the scenarios below. You can find required instructions in the followinf articles: *Stopping IWTM System*, *Stopping PostgreSQL* and *Restoring Data Storage*.

If you use Data Storage and it is not damaged then no action with the storage is required when restoring DB. Data Storage does not interact with the database directly.

To restore only DB:

1. Stop the Traffic Monitor services.
2. Restore the DB the chosen way.
3. Start the Traffic Monitor services.

To restore only Data Storage:

1. Stop the Traffic Monitor services.
2. Stop the database.
3. Restore Data Storage.
4. Start the database.
5. Start the Traffic Monitor services.
6. Make sure that Traffic Monitor is connected with Data Storage. If necessary, configure connection. For detailes, *Configuring Connection with Data Storage*.

To restore both DB and Data Storage:

1. Stop the Traffic Monitor services.
2. Restore Data Storage.
3. Restore the database.
4. Start the Traffic Monitor services.
5. Make sure that Traffic Monitor is connected with Data Storage. If necessary, configure connection.

> **Important!**
>
> If you need to update Traffic Monitor to a new version, you should first restore the database. Updating Traffic Monitor either before or during the database recovery is not supported.

Restoring to the Same Database

Below you can find instructions on how to restore data to a database that has the same directory structure as the one that was backed up.

To restore the database without creating a new one:

1. Make sure the database environment is healthy. Check the existing database, the database server that hosts the existing database, and the computer that hosts the database server.
2. Stop PostgreSQL services:
   ```
   systemctl stop postgresql-15
   systemctl stop pgagent-15
   ```
3. Take the following steps:
   a. Delete all the content of the **/u01/postgres/**, **/u02/pgdata/**, **/u02/pgdata1/**, **/u02/arch/** directories;
   b. Copy the content of the **/opt/IWTM_Backup_Files/postgres** directory to the **/u01/postgres/** directory;
   c. Copy the content of the **/opt/IWTM_Backup_Files/pgdata/** directory to the **/u02/pgdata/** directory;
   d. Copy the content of the **/opt/IWTM_Backup_Files/pgdata1/** directory to the **/u02/pgdata1/** directory;
   e. Copy the content of the **/opt/IWTM_Backup_Files/arch/** directory to the **/u02/arch/** directory.
4. Check the permissions. If necessary, change them:
   ```
   chown -R postgres /u01/postgres/
   chown -R postgres /u02/pgdata/
   chown -R postgres /u02/pgdata1/
   chown -R postgres /u02/arch/
   ```
5. Start the database:
   ```
   systemctl start postgresql-15
   systemctl start pgagent-15
   ```

Restoring to a New Database

To restore the database by creating a new one:

1. Install a new database and configure it according to the following article: *InfoWatch Traffic Monitor. Installation Guide. Configuring Remote Database*.

   > **Note:**
   >
   > If you have installed the DB on the same server, reinstall additional packages `pgagent` and `postgresql-contrib` during configuration by using the `--reinstall` parameter in the command.

2. Copy the content of the directories, check permissions and start the database services similarly to *Restoring to the Same Database*.
   You should copy DB files to the same directories, in which they were stored in the old database.

Restoring Data Storage

Restoring the storage of binary data lies in restoring buckets from their backups. This procedure consists of the following steps:

1. Getting a token to work with the Data Storage service
2. Creating the task of restoring bucket
3. Checking successful completion of the task

When sending **curl** requests, you can specify the following values as a server address `<ADDRESS>`:

- **localhost:40731**, if you send requests from the server where Data Storage is installed.
- **<server_IP_address>:40732**, if you send requests from another server.

Getting a token to work with the Data Storage service

To work with the Data Storage service, you need to get a Bearer token for HTTP authorization. In requests to Data Storage, specify it in the Authorization header.
The token value can become outdated. So in response to a request, the service returns the 440 HTTP code. In this case, get the token again.

To get the token, send the following request to Data Storage:

```
curl -X GET -I --silent "http://<ADDRESS>/v2/logon?extAPIToken=<DS_user_token>" 2>&1 | grep
'^Authorization'
```

where `<ADDRESS>` is the server address. For details, see above.
`<DS_user_token>` is the Data Storage authorization token. The default value: `1exwo92iwpczl1hy6jxy`.
To learn more, see *Configuring Connection with Data Storage*.

As a result, you will see the Authorization header in the following format:
`Authorization: Bearer <token_value>`

Creating the task of restoring bucket

To create a task of restoring bucket, send the following request:

```
curl -X POST --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/restores?bucketID=<BUCKET_ID>"
```

where `<AUTH_HEADER>` is the Authorization header;
`<ADDRESS>` is the server address;
`<BUCKET_ID>` is the bucket identifier.

In response, the ID of the created task will be displayed:

```
{
  "taskID": "fe8163f2-a0fb-4990-8bb7-87aee5c425bc"
}
```

Also you can create the task of restoring bucket and get the ID of this task using the **jq** utility:

```
curl -X POST --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/restores?bucketID=<BUCKET_ID>"
 | jq --raw-output '.taskID'
```

Next, you can check successful restoration of bucket using the task ID.

> **Important!**
>
> You can restore the bucket only if there is no one with the same ID in Data Storage.

Checking successful completion of the task

To make sure that restoration completed successfully, check the status of the task. For this, send the request:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/restores?taskID=<TASK_ID>"
```

where `<AUTH_HEADER>` is the Authorization header;
`<ADDRESS>` is the server address.;
`<TASK_ID>` is the task identifier.

If succesfull, you will see the following response:

```
{
  "bucketID": "foobucket",
  "taskStatus": "Success"
}
```

Make sure that the task is completed and has one of the following statuses: Success, Canceled, Failed.

Also you can get the task status using the **jq** utility:

```
curl -X GET --header "<AUTH_HEADER>" --silent "http://<ADDRESS>/v3/buckets/restores?taskID=<TASK_ID>" | jq
--raw-output '.taskStatus'
```

8.2.5 Database Server Routinely Maintenance

> **Warning!**
>
> We highly recommend not shutting down the DB server with the power button: in certain cases, this may cause the DB corruption.

If you need to carry out routine maintenance on the database server, take the following steps:

1. Close all instances of the Management Console. Make sure that all connections of other programs to Traffic Monitor database scheme are closed; if there are any active connections, disable them.
2. Stop all TM processes on Traffic Monitor server:
   `iwtm stop`
   `systemctl stop iwtm-php-fpm`
   `systemctl stop nginx`
3. On the database server, retrieve the list of scheduled jobs. For this, in psql, run the following query as the database scheme owner:
   `select jobname,  case when jobagentid is null then 'scheduled' else`

```
'running' end state
from pgagent.pga_job
where jobenabled= true;
```

4. Disable jobs by running the following script:
   ```
   select iwtm.pkg_utility_disable_job('JOB_1');
   ```
   ...
   ```
   select iwtm.pkg.utility_disable_job('JOB_N');
   commit;
   ```
   where `JOB_1… JOB_N` are the names of jobs to be disabled.
5. To make sure that there are no running jobs, run the following query:
   ```
   select pga_job.jobname
   from pgagent.pga_job
   where pga_job.jobagentid is not null;
   ```
   A current task cannot be stopped in PostgreSQL, this should be done by stopping the agent from Linux using thy following command:
   `systemctl stop pgagent-15` (stopping all tasks with one command)
6. Make all the necessary operations with the database.
7. After all operations are complete, check the connection to the database server from Traffic Monitor server:
   ```
   psql postgres iwtm
   ```
   where `server_name` is database name or IP address
   If the test is successful, the following psql invitation will appear:
   `postgre=#` or `postgres=>`
8. Run the Traffic Monitor server processes:
   ```
   iwtm start
   systemctl start iwtm-php-fpm
   systemctl start nginx
   ```
9. Check if the system log contains errors. The path to the system log:
   ```
   /var/log/messages
   ```
10. If the system log contains any error messages, consult the technical support service.
11. Enable running previously disabled jobs:
    ```
    BEGIN
    select iwtm.pkg_utility_enable_job('JOB_1');
    …
    select iwtm.pkg_utility_enable_job('JOB_N');
    commit;
    end;
    /
    ```
    where `JOB_1… JOB_N` are the names of the previously stopped jobs.

8.2.6 Database logging

Traffic Monitor saves logs of PostgreSQL database to `/u01/postgres/pg_log/postgresql.log` file. The System also generates logs on tasks running with pgAgent and saves them to `/u01/postgres/pg_log/pgagent-15.log` file.

PostgreSQL DBMS operation in Traffic Monitor is logged with a script. This script allows you to set the logging level and export log data to a CSV file. The script uses settings of the current configuration file: `/opt/iw/tm5/csw/database.conf`.

The script supports the following commands:

- To set the logging level, run the command:
  `/opt/iw/tm5/csw/postgres/scripts/sys_log/sys_log.sh level <level_value>`
  where `<level_value>` can be `debug`, `info`, `warning` or `error`.
  If you run the command without specifying the level value, the current logging level will be displayed.
- To export the log archive, run the command:
  `/opt/iw/tm5/csw/postgres/scripts/sys_log/sys_log.sh export`
  If necessary, you can set the number of lines to be exported:
  `/opt/iw/tm5/csw/postgres/scripts/sys_log/sys_log.sh export <number_of_lines>`
  The exported file gets a name of `iwtm_sys_log_<yyyy-mm-dd_hh-mm-ss>.csv.gz` format and is saved to the current directory.

On how logs are stored on the disk, see *Rotation of database logs*.

Rotation of database logs

Rotation of database logs is the process of archiving, storing, and deleting obsolete data in logs. This process is required to save space and free up disk space. In Traffic Monitor, rotation applies to the DB log in a text format: `/u01/postrges/pg_log/*.log`. This log file is rotated by the **logrotate** service. Rotation applies to each 100 MB of the file. When the number of files exceeds 100, the log file is deleted.

Settings of the **logrotate** service are stored in `/etc/logrotate.d/iwtm-pg-conf`.

9 Administrating the Server End of InfoWatch Traffic Monitor

This chapter describes Traffic Monitor server end components, their settings and how to use them:

- Traffic Monitor Server Processes;
- Configuring OCR;
- Configuring Settings for Processing Enclosed Archives;
- Configuring Settings of Search Using Wildcards;
- Configuring Integration with Data Analysis Service;
- Configuring Connection with Data Storage;
- Logging System Work;
- File Queues;
- Recovering the System after Failures;
- Managing Languages Including Morphological Support;
- Configuring Forwarding Data to SIEM;
- Deleting Temporary Files.

9.1 Traffic Monitor Server Processes

This section covers:

- List of Traffic Monitor Server End Processes;
- Setting Traffic Monitor Server End Configuration Files;
- Using Traffic Monitor Server End Processes.

9.1.1 List of Traffic Monitor Server End Processes

The System works by means of running processes: the same process can run simultaneously in several instances.

Purpose	Process name	Process description	Configuration file
Data collection	`iw_icap`	Processes HTTP traffic. Receives HTTP requests from the ICAP client. Extracts data from HTTP requests. Then, the extracted data is added to the XML context. The XML context is sent to the analysis subsystem for the check. After the analysis, the response is sent to the ICAP client to allow/deny the delivery of the HTTP request. The HTTP request is passed to the **`iw_x2x`** process through the file queue, and then to the **`iw_x2db`** process.	`/opt/iw/tm5/etc/icap.conf`
	`iw_icap_buf`	Saves parts of a large file in its storage. Then, sends the created object as a complete file with a single request to **`iw_icap`** for analysis.	`/opt/iw/tm5/etc/icap_buf.conf`
	`iw_proxy`	Receives a copy of the traffic, divides it to HTTP, ICQ, and SMTP traffic, and sends it to the `iw_messed` module. Includes the following processes:	`/opt/iw/tm5/etc/proxy.conf`

Purpose	Process name	Process description	Configuration file
		• `iw_proxy_http` – receives a copy of HTTP traffic. Receives an HTTP request, generates XML context from the received object. Then, XML context is transferred to the analysis and decision making subsystem **`iw_analysis`**. • `iw_proxy_icq` – receives a copy of ICQ traffic. Receives an ICQ message, generates XML context from the received object. Then, XML context is transferred to the analysis and decision making subsystem **`iw_analysis`**. • `iw_proxy_smtp` – receives a copy of SMTP traffic. Receives an SMTP message, generates XML context from the received object. Then, XML context is transferred to the **`iw_messed`** analysis subsystem. After the analysis is complete, the object is passed to the **`iw_x2x`** process and then to the **`iw_x2db`** process.	
	`iw_smtpd`	Receives SMTP messages. When integrated with the Postfix mail server, it receives incoming messages from Postfix. If there is no integration with Postfix, it receives incoming messages from the corporate mail server or mail client (depending on the settings of your mail system). Receives incoming messages in SMTP format, converts SMTP envelope data to the XML context. Then it transfers SMTP message and XML context to the **`iw_messed`** process.	`/opt/iw/tm5/etc/smtpd.conf`
	`iw_capstack`	Processes traffic transferred by POP3, IMAP, and NRPC protocols.	`/opt/iw/tm5/etc/capstack.conf`
	`iw_messed`	Processes SMTP messages. Extracts data from SMTP, POP3, and IMAP objects. The extracted data is then added to the received XML context. The ready XML context (containing the data extracted from the envelope and the email) is transferred to the analysis and decision making subsystem. After the analysis, **`iw_messed`** transfers the allowed SMTP messages to the mail system component that delivers mail (only in the normal and transparent transport modes). If integration with the Postfix mail server is	`/opt/iw/tm5/etc/messed.conf`

Purpose	Process name	Process description	Configuration file
		configured, then this component is Postfix. If there is no integration with Postfix, then the component depends on the settings configured in your mail system: it can be a corporate mail server or a mail client. The checked SMTP email is passed to the **iw_x2x** process via the file queue, and then to the **iw_x2db** process.	
	iw_xapi	Includes 2 services: **iw_xapi_xapi** and **iw_xapi_puppy**. Receives objects from Device Monitor and external systems (through adapters) via the Thrift interface and places them in a file queue. Then it sends objects to the **iw_analysis** process. When receiving EML objects, sends them to the **iw_messed** process.	/opt/iw/tm5/etc/xapi.conf
	iw_analysis	Takes objects from the file queue, previously placed there by **iw_ xapi_xapi** / **iw_xapi_puppy**, **iw_proxy_icq**, **iw_proxy_http**. By means of the file queue it transmits objects to **iw_warpd**, **iw_cas**, **iw_pas**, **iw_luaengined** for processing. Then it sends the objects to the file queue for sending to the **iw_x2x** process, and then to the **iw_x2db** process.	/opt/iw/tm5/etc/analysis.conf
	iw_rest2push	Allows receiving objects from external systems via the HTTP/HTTPS protocol. Connects to the iw_xapi_puppy service, sends events via the Thrift protocol.	/opt/iw/tm5/etc/rest2push.conf
Data processing	iw_warpd	Manages processes of extracting data from containers enclosed in intercepted objects.	/opt/iw/tm5/etc/warpd.conf
	iw_image2text_fre	Recognizes text in images using ABBYY FineReader OCR.	/opt/iw/tm5/etc/image2text_fre.conf
	iw_image2text_ts	Recognizes text in images using Tesseract OCR.	/opt/iw/tm5/etc/image2text_ts.conf
	iw_cas	Takes the role of the content analysis server. Interception subsystems send text requests in plain text format to the **iw_cas** process for content analysis. When the analysis is complete, the request result is sent to the analysis and decision-making subsystem.	/opt/iw/tm5/etc/cas.conf

Purpose	Process name	Process description	Configuration file
	`iw_image_autoling`	Takes the role of the graphic objects automatic classifier.	`/opt/iw/tm5/etc/iw_image_autoling.conf`
	`iw_text_autoling`	Takes the role of the text objects automatic classifier.	`/opt/iw/tm5/etc/iw_text_autoling.conf`
	`iw_pas`	Receives analysis results from **`iw_cas`** . Defines the presence of a protected object and adds corresponding attributes to the object.	`/opt/iw/tm5/etc/pas.conf`
	`iw_luaengined`	Enables running of a LUA script according to the actual policy.	`/opt/iw/tm5/etc/luaengined.conf`
	`iw_x2x`	The process gets data with XML+DAT files via the file queue. The files provided are modified and sent to the **`iw_x2db`** process.	`/opt/iw/tm5/etc/x2x.conf`
	`iw_tech_tools`	Verifies conditions for DB unloadings and regular expressions, makes it possible to normalize text according to regular expressions.	`/opt/iw/tm5/etc/tech_tools.conf`
Loading to DB	`iw_qmover_client`	Works on the Traffic Monitor server installed in a branch office. Sends intercepted objects to the headquarters' DB.	`/opt/iw/tm5/etc/qmover_client.conf`
	`iw_qmover_server`	Works on the Traffic Monitor server installed in the central office. Gets objects received from the Agent installed in the branch office. Sends objects to the **`iw_x2db`** process through **`iw_x2x`** for saving to the DB.	`/opt/iw/tm5/etc/qmover_server.conf`
	`iw_x2db`	Loads objects verified by the analysis and decision making systems from the output file queue of the **`iw _x2x`** process to the DB.	`/opt/iw/tm5/etc/x2db.conf`
Infrastructure	`iw_adlibitum`	Manages processes of receiving actual data from the Active Directory, Domino Directory, Samba DC, Astra Linux Directory, Astra Linux Directory Pro, and FreeIPA servers.	`/opt/iw/tm5/etc/adlibitum.conf`
	`iw_agent`	Required for managing the System configuration.	`/opt/iw/tm5/etc/agent.conf`

Purpose	Process name	Process description	Configuration file
	`iw_attributor`	Enables interaction with Data Analysis Service: transfers data for analysis and loads the received data to the DB.	`/opt/iw/tm5/etc/attributor.conf`
	`iw_blackboard`	Provides a Thrift interface to read a file queue of commands and delegates their execution to other services. The commands include adding statuses and contacts to persons, updating meta information about contacts, adding statuses to workstations, sending notifications about events, adding applications to lists.	`/opt/iw/tm5/etc/blackboard.conf`
	`iw_bookworm`	Acts as a reference book in the System.	`/opt/iw/tm5/etc/bookworm.conf`
	`iw_cas_config_compiler`	Converts the configuration file of the content analysis server to a binary file for using the configuration in the content analysis.	`/opt/iw/tm5/etc/cas_config_compiler.conf`
	`iw_configerator`	Generates configuration that is sent to Device Monitor.	`/opt/iw/tm5/etc/configerator.conf`
	`iw_deliver`	Delivers emails. Sends SMTP messages to recipients if the delivery is allowed in the Management Console (only in normal and transparent transport modes). Besides, it delivers SMTP messages if, for some reason, they have not been delivered by the **`iw_messed`** process (for example, if there is no connection with the mail relay server or a mail client).	`/opt/iw/tm5/etc/deliver.conf`
	`iw_metainfo_fetcher`	Selects data from the DB containing meta information about all contacts, including contacts participating in events (for indexing). Sends them to the **`iw_is`** process.	`-`
	`iw_audit_fetcher`	Selects extended audit events from the DB for indexing. Sends them to the **`iw_is`** process.	`-`
	`iw_is`	Manages the data indexing process and the operation of search servers for full-text search. Prepares the files with data description, which get to the indexer through the file queues, and starts the indexer.	`/opt/iw/tm5/etc/is.conf`
	`iw_indexer`	Indexes the text in events from the DB. Gets access to the DB for indexing it and adds indexes to the file storage. When a **`sphinx`** search query	`/opt/iw/tm5/etc/indexer.conf`

Purpose	Process name	Process description	Configuration file
		is running, receives object IDs from the file storage of indexes.	
	`iw_kicker`	Enables operation of WebGUI and starts the following services: `agent`, `blackboard`, `export`, `import`, `report`, `notifier`, `selection`, `systemcheck`, `xapisamplecompiler`, `samplecompiler`, `querytracker`, `reporttracker`	`/opt/iw/tm5/etc/kicker.conf`
	`iw_licensed`	Licensing subsystem. Monitors and processes licenses installed in the System.	`/opt/iw/tm5/etc/licensed.conf`
	`iw_rammer`	Sends emails with processing errors when working inline.	`/opt/iw/tm5/etc/rammer.conf`
	`iw_sample_compiler`	Creates fingerprints from uploaded sample files.	`/opt/iw/tm5/etc/sample_compiler.conf`
	`consul`	Detects, registers, and monitors availability of services which interact with the user interface.	`/opt/iw/tm5/etc/consul/consul.json`
	`iw_system_check`	Collects data from the Nagios service and provides it for display in the Management Console.	`/opt/iw/tm5/etc/system_check.conf`
	`iw_updater`	Loads the configuration from the DB to the Traffic Monitor server.	`/opt/iw/tm5/etc/updater.conf`
	`iw_space_manager`	Manages tablespaces in the **Tablespaces** section of the Management Console.	`/opt/iw/tm5/etc/space_manager.conf`

9.1.2 Setting Traffic Monitor Server End Configuration Files

You can find the list of System configuration files and their settings that can be changed to configure the System for operation in your organization in the *InfoWatch Traffic Monitor. Configuration Files Guide* document. We recommend not editing files that are not described in this document.

Configuration files are located in `/opt/iw/tm5/etc`.

The System configuration files are in JSON format. Configuration file names match the names of corresponding System components to be configured. For example, for configuring **iw_x2x** component, **x2x.conf** configuration file is used**.**

There are common configurable settings and parameters specific for each configuration file

See the document *InfoWatch Traffic Monitor. Configuration Files Guide:*

- common configurable settings:
  - *General section Bookworm;*
  - *General section Discovery;*
  - *General section Logging;*
  - *General section Statistics;*
  - *General section Statistic;*
  - *General section ThriftServers.*
- parameters specific for each configuration file.

9.1.3 Using Traffic Monitor Server End Processes

You can find the list of Traffic Monitor processes in *List of Traffic Monitor Server End Processes*.

All Traffic Monitor Server processes are launched under `iwtm` user, which is created automatically during Traffic Monitor Server installation (for more information on user accounts, see *InfoWatch Traffic Monitor. Installation Guide*).

> **Note:**
>
> The user name is registered in unit files of processes in `User` parameter. For correct operation of the System, we recommend leaving the value of this parameter unchanged.

During Traffic Monitor Server operation, the state of processes is checked by `pguard` script which is created in `/opt/iw/tm5/bin` directory during the installation. If for some reason, one or more Traffic Monitor Server processes are stopped, **pguard** script restarts them.
Traffic Monitor Server processes are managed with bash script `/opt/iw/tm5/bin/iwtm` which is an autorun script for run levels 2, 3, 4, 5.

> **Note:**
>
> - The `iw_qmover_server`, `iw_qmover_client` processes are available, however, autorun is disabled for them by default. Each process is launched on its end (`iw_qmover_client` in branches and `iw_qmover_server` in the central office.
> - The `iw_rammer` process is not installed by default. After manual installation, the process is available in the process list.
> - In case you have to stop services on servers of a distributed Traffic Monitor cluster, first stop processes on a server installed with the *Interceptors* template, and at the end stop processes on a *Database* server. To start processes, begin with a server installed with the *Database* template and finish with *Interceptors*.
> - If Traffic Monitor Server uses Data Storage, actions for stopping/starting processes are the same as without the storage.

Start keys for `iwtm` script:

Start key	Purpose
`start`	Starting processes

Start key	Purpose
`stop`	Safe stop of processes
`status`	Displaying process state (running/not started/available/ unavailable)
`restart`	Restarting processes (sequential execution of start+stop commands)
`reload`	Reloading process configuration
`kill`	Transmitting a signal to immediately end the process (SIGKILL)
`test`	Displaying all available iwtm daemons
`enable`	Setting the "available" status to a daemon - the daemon autostart is allowed
`disable`	Setting the "unavailable" status to a daemon - the daemon autostart is forbidden
`services_state`	Displaying all available iwtm daemons with bit masks

Sample commands

Command	Action
`iwtm status`	Check the state of Traffic Monitor Server processes

One of the following commands:
`service iw_cas start` (if you need to view logs)
`iwtm start cas`
`iwtm start iw_cas`
`iwtm start iw_cas.service`
`systemctl start iw_cas` (viewing logs is unavailable)
`systemctl start iw_cas.service`
	Start the content analysis server (CAS)

Warning!

PID files are stored in `/opt/iw/tm5/run` directory. In case of an emergency shutdown, you should delete all PID files before starting processes.

Changing process autorun parameters

You can find information on enabling/disabling autorun for processes in *Check of Process Autorun*.

For each process managed by **iwtm** bash script, autorun parameters can be specified in a command line. Enabling/disabling autorun for processes (with `iw_sniffer` service as an example) is described in *Enabling and Disabling Process Autorun*. For details about systemd commands see article *Commands to work with systemd*.

9.2 Configuring OCR

The OCR technology allows you to convert hand-written, typed or printed text to text data. By default, the technology is disabled for all services, event types, and related protocols. OCR is used for analysis of:

- Intercepted images;
- Images loaded as sample documents.

The following processes are used for the technology operation:

- **`iw_warpd`** extracts data from containers included into intercepted objects;
- **`iw_sample_compiler`** creates digital prints from downloaded sample documents.

Using the configuration files of these processes, you can configure OCR technology usage. For details, see *Configuring OCR Extractors*.

If necessary, you can configure OCR to analyze intercepted images. The following methods of configuring OCR are available:

Interception level	Settings apply to the content of events	Priority of the setting
Service	Received via protocols of event types which belong to this service (if OCR settings are not specified for these event types or protocols)	Low
Event type for a particular service	Received via protocol of this event type (if OCR settings are not specified for these protocols)	Medium
Protocol for a particular event type	Received via the selected protocol for the specified event type	High

OCR settings are specified in the directory of **`iw_bookworm`** daemon, in `/opt/iw/tm5/etc/config-perm/bookworm` catalogue:

- `ocr.xml` – preset settings;
- `ocr_custom.xml` – custom settings for a specific type of implementation.

Important!

When you update the System to current version, OCR parameters are reset to preset ones. In case you add or change OCR functional capacity, the new settings to be available within the `ocr.xml` file. User settings remain within the `ocr_custom.xml` file.

9.2.1 Enabling OCR technology when implementing the System

You can enable using OCR technology when you implement the System. For this, modify the `ocr_custom.xml` file (for details see *Configuration File ocr_custom.xml*) the following way:

1. Select required objects: services, protocols, and event types[1].
2. Set identifiers for service directory objects and event types (files with a description are located in the `/opt/iw/tm5/etc/config/bookworm/` catalogue).
3. Enable OCR (`ocr_enabled="true"`).

> **Note:**
>
> [1] New event types registered via plugins are added to `/opt/iw/tm5/etc/config/bookworm/services.xml` . They can be processed or moved to the `ocr_custom.xml` file.

9.2.2 Configuring OCR on several nodes

If you need to enable OCR on several server instances (for a distributed installation), you should set rules for each instance. For this:

1. In catalogue `/opt/iw/tm5/etc/config-perm/bookworm` , create a new XML file that will describe the node.
2. Specify the created XML file in `CustomNodeXMLPath` section of configuration file `/opt/iw/tm5/etc/bookworm.conf` .
3. In `<ocr_option node>` parameter, specify instance name from `/opt/iw/tm5/node_name` file.
4. Define overridable rules for the instance as described above.

> **Note:**
>
> In this case, settings have the following priority (in descending order):
>
> - Catalogue with settings for the server instance;
> - Catalogue with custom settings;
> - Catalogue with preset settings.

9.2.3 Configuration File ocr_custom.xml

Below you may find disabled OCR rules, where:

- `object_type` – object type;
- `service` – services;
- `protocol` – protocols;
- `key` – keys;
- `mnemo` – catalogues objects IDs of object types, services and protocols.

When enabling the rules, it is recommended specifying only the required ones:

```
<ocr_options node="*" ocr_enabled="false">
  <service key="CE6D8E0E27DA11E2962FC1DB6088709B00000000" mnemo="email" ocr_enabled="false">
    <object_type key="D2B5132E27DA11E28444C2DB6088709B00000000" mnemo="email" ocr_enabled="false">
      <protocol key="501A9868F3460E2E5AAF0C6AB21F8D6F63C09109" mnemo="pop3" ocr_enabled="false"/>
      <protocol key="2ECB2A264E31677895BC5D8845D33BC26F637CDF" mnemo="imap" ocr_enabled="false"/>
      <protocol key="A324CCB227DA11E2A8D99FDB6088709B00000000" mnemo="smtp" ocr_enabled="false"/>
      <protocol key="5F7E60DB3E86EFF90FDA87E72209EA1B0E017F16" mnemo="mapi" ocr_enabled="false"/>
      <protocol key="93B79EB44EE1A45522C08319EC47B499294776D7" mnemo="nrpc" ocr_enabled="false"/>
    </object_type>
    <object_type key="D8333C9027DA11E29E34CADB6088709B00000000" mnemo="email_web" ocr_enabled="false">
      <protocol key="A779DB3627DA11E2ADDCA0DB6088709B00000000" mnemo="http" ocr_enabled="false"/>
      <protocol key="7ED97C84BDBDD99C1C21AC0A6D6191F6A891C440" mnemo="https" ocr_enabled="false"/>
    </object_type>
  </service>
  <service key="DD6ECB5227DA11E2B507CBDB6088709B00000000" mnemo="im" ocr_enabled="false">
    <object_type key="E266719627DA11E2A83ECCDB6088709B00000000" mnemo="im_icq" ocr_enabled="false">
      <protocol key="ABA5D02027DA11E2B78FA1DB6088709B00000000" mnemo="oscar" ocr_enabled="false"/>
    </object_type>
    <object_type key="EEC2D87627DA11E29EA6D2DB6088709B00000000" mnemo="im_mail_ru" ocr_enabled="false">
      <protocol key="B8C7FC6A27DA11E2A080B7DB6088709B00000000" mnemo="mmp" ocr_enabled="false"/>
    </object_type>
    <object_type key="F249A80827DA11E29BA2D3DB6088709B00000000" mnemo="im_skype" ocr_enabled="false">
      <protocol key="DD3D593857521DE2E9618C26FD3E1B914F8A16F9" mnemo="skype" ocr_enabled="false"/>
    </object_type>
    <object_type key="E7C3A26C27DA11E296D3CDDB6088709B00000000" mnemo="im_xmpp" ocr_enabled="false">
      <protocol key="AF69EA3427DA11E2BE09A2DB6088709B00000000" mnemo="xmpp" ocr_enabled="false"/>
    </object_type>
  </service>
  <service key="62E7C68AD1354D118282FAFF07DA59ED00000000" mnemo="multimedia" ocr_enabled="false">
    <object_type key="55E10E81F5C94B3FB742CB61CA9476F800000000" mnemo="multimedia_photo"
ocr_enabled="false"/>
  </service>
  <service key="0794BBB227DB11E2BD81E9DB6088709B00000000" mnemo="web" ocr_enabled="false">
    <object_type key="0CAFCC9027DB11E2AD27EDDB6088709B00000000" mnemo="web_common" ocr_enabled="false">
      <protocol key="A779DB3627DA11E2ADDCA0DB6088709B00000000" mnemo="http" ocr_enabled="false"/>
      <protocol key="7ED97C84BDBDD99C1C21AC0A6D6191F6A891C440" mnemo="https" ocr_enabled="false"/>
    </object_type>
  </service>
  <service key="111CBA0427DB11E2AB82EEDB6088709B00000000" mnemo="file" ocr_enabled="false">
    <object_type key="1515A7B027DB11E2BBB2EFDB6088709B00000000" mnemo="file_exchange" ocr_enabled="false">
      <protocol key="BEFB1A7C27DA11E2AB10B8DB6088709B00000000" mnemo="ftp" ocr_enabled="false"/>
    </object_type>
    <object_type key="190D004827DB11E287B1F0DB6088709B00000000" mnemo="file_copy_out" ocr_enabled="false"/>
  </service>
  <service key="1DA8A90427DB11E289E8F5DB6088709B00000000" mnemo="print" ocr_enabled="false">
    <object_type key="225BD8F427DB11E2926DF9DB6088709B00000000" mnemo="print_common" ocr_enabled="false"/>
  </service>
  <service key="7843FC5BEA024E9B274E26DB43B7E680D8BC9356" mnemo="placement" ocr_enabled="false">
    <object_type key="602A224D9335579214E3188D1D2745DB9F85D500" mnemo="crawler" ocr_enabled="false"/>
  </service>
</ocr_options>

<ocr_options node="*" ocr_enabled="false">
</ocr_options>
```

Note:

To enable OCR on all the levels at one time (service level, event type and protocol level), at the end of the `ocr_custom.xml` file replace the line:

```
<ocr_options node="*" ocr_enabled="false">
```

with the following:

```
<ocr_options node="*" ocr_enabled="true">
```

9.3 Configuring Settings for Processing Enclosed Archives

The attachment processing is configured in the following configuration files:

- `/opt/iw/tm5/etc/`**`extractors.conf`** (see *InfoWatch Traffic Monitor. Configuration Files Guide. extractors.conf*);
- `/opt/iw/tm5/etc/config-perm/bookworm/`**`extractors.xml`** file (see *The extractors.xml Configuration File*);
- `/opt/iw/tm5/etc/config-perm/bookworm/`**`extractors_customization.xml`** file (see *The extractors_customization.xml Configuration File*).

Important!

If MS Exchange uses a journal rule, then, for the correct processing of such emails, enable the corresponding option in Traffic Monitor. For this, in the `messed.conf` configuration file, specify:

```
"ExchangeJournalReports": true,
```

To make the option active, reload the configuration, or restart the service.

9.3.1 The extractors.xml Configuration File

The `/opt/iw/tm5/etc/ config-perm/bookworm/`**`extractors.xml`** file contains directory settings and signatures base which define file types when unpacking archives and enclosures.

Parameter	Description
`FilenameCharsets`	If file name encoding differs from the standard one (ANSI, UTF) then the filename processes as corresponding to encodings listed in this setting. Several values are listed comma separated; the System sequentially tries to process the filename in listed encodings while it finds a matched one. The default values are `UTF-8`, `cp1251`, `cp866`

`MinFileSizeInBytes`	Minimum file size (in bytes) subjected to unpacking. This limit lets increase the System performance due to refusing of small files unpacking. The default value is `10` **Note:** The setting also applies to *Clipboard* and *Keylogger* events, i.e. the System does not process clipboard/keyboard text if its volume is less than the value specified.
	Maximum size of file whose signature is not defined. Exceeding the threshold value file is automatically considered to be binary and is not sent to `iw_cas`. The default value is `25600`
`SpeedInKBs`	Proposed extractor's operation speed per 1 GHz of the processor (in Kb/s). Can take integer values from 1 and greater. The default value is `100`
`TimeoutInSec`	Awaiting time till file processing ends (in seconds). The default value is `0`
`UseLog`	Logging extractor. The default value is `false`

If the general values of parameters, which are specified in the `<extractor_defaults>` section, are not appropriated for some file types then enable these parameters in sections of needed file types with other values.

Example:

The extractor operating time can be calculated by the formula:

The extractor operating time = File size in KB / 1024 / (`SpeedInKBs` * CPU clock speed in MHz / 1000)

Other sections define signatures used for detection of file types from archives and enclosures. Each section contains the same set of parameters:

Parameter	Description
`Extension`	Extension of archive files. Optional parameter which is needed for correct operation of some archiving utilities requiring correct extensions of incoming files **Example:** `Ext = arj`
`Name`	Format name. Used for exact detection of format name when file unpacking. To define detection/detailing of format the setting can include a list of names:

Parameter	Description
	" `defaultname, name1(code1), name2(code2)` " In this case the format can be defined basing on the code (`codeN`) assigned to the file when unpacking. For example, if the unpacking is ended with `code1` , the file is considered as having `name1` format. The list of names can include default name (`defaultname`) which will be assigned when the code is not assigned: • **Example 1** (one format name): `text/xml` • **Example 2** (list of format names): `"application/arj, UNKNOWN(9), text/encrypted(4)"` By default, the file format is defined as `application/arj` . Two others for specific cases are: unknown format (`UNKNOWN`), encrypted format (`text/encrypted`)
`Signature`	Format signature which is 'standard' sequence of values (bytes) with initial displacement and masks. This sequence is needed for unambiguous definition of files format by the content of those files. The signature can include up to 16 triades of the following format: `[OFFSET]:BYTE_VALUE/[MASK]` There can be string signature like " `BZh` " for BZIP2. This signature means specified bytes sequence from 0 displacement. The search of string signatures is not case-sensitive. The `OFFSET` setting is optional. By default, it equals 0 for the first byte and increases by 1 for each later one. The displacement can be specified using ' `?` ' symbol – searching of the pattern in first N bytes of a file. For example, to search pattern in first 500 bytes, you should specify ' `500?` ' displacement. The `MASK` setting is optional and equals to `0xFF` by default. **Example** (signature of a heading of WAV format): `Sign = "'RIFF', 0x8: 'WAVEfmt'"` Starting from 0 displacement, there is a sequence ' `R` ',' `I` ',' `F` ',' `F` ', then from 8 displacement there should be ' `W` ',' `A` ',' `V` ',' `E` ',' `f` ', ' `m` ',' `t` '
`Comment`	The explanation commentary **Example** (indicating the version of archiving utility): `Comment = "bzip2 v. 1.0.x"`
`Command`	Command string for file extracting **Example:** `Extract = "/usr/bin/bzip2 -f -dc ${SRC} > ${OUTDIR}/${SRC}"` Where `${OUTDIR}` is a catalogue where to the archive will be extracted and `${SRC}` is a name of file to unpack.

Parameter	Description
`ExcludeFromContext`	If the parameter is enabled (`true`) then the file is unpacked but information on archiving utility is not added to object's XML-context. This lets to exclude fictional containers like MS-TNEF from object's context. By default, the parameter is enabled.
`CaseInsensitive`	Indicates whether file processing is case-sensitive. The default value is `false`
`CreateTags`	Indicates whether to extract metadata from a file. The default value is `true`
`Archive`	Regulates the kind of retrieved information: • `true` – files; • `false` – data.
`PIRegexp`	Contains a description of PIRE signature for file format detection. Can be specified several times within the same `Extractor` . Includes the following attributes: • `assurance` – the accuracy of detecting file format with this signature; • `offset` – offset where to start searching for the signature; • `max_length` – the length of the section to be checked; • `case_sensitive` – enables case sensitivity in the signature description.
`DetectableByRegexp`	Allows you to disable format detection by means of PIRE (instead, signatures from `Signature` will be used).
`OpenFifoRetryTimeout`	Timeout during few attempts to open FIFO queue
`OpenFifoRetryCount`	Number of attempts to open FIFO queue
`ResultCharset`	Specifies the code of extracted text.
`NeedProcess`	If the files with MIME type specified should be sent for analysis.

Note:

Some intercepted files of XML format may contain sensitive data specified within XML tags of file markup. By default, the System does not extract text from tags. To configure text extraction from XML tags:

1. Open `/opt/iw/tm5/etc/config-perm/bookworm/extractors.xml` file for editing.
2. Find the element with the following contents:

```
<Command>bin/text-extractor -t xml -i ${OUTDIR}</Command>
```

3. Replace **`-t xml`** parameter with **`-t xmltxt`**.
4. Save changes and close the file.
5. Reload configuration of **`iw_bookworm`** process by running the command:

```
iwtm reload bookworm
```

6. Restart the processes:

```
iwtm restart analysis
iwtm restart messed
iwtm restart icap
```

As a result, the extracted XML text contains tags and other service information, which is not saved by default text extraction method for XML.

9.3.2 The extractors_customization.xml Configuration File

In `/opt/iw/tm5/etc/config-perm/bookworm/`**`extractors_customization.xml`** file, you can add or adjust extractor settings specified in `/opt/iw/tm5/etc/config-perm/bookworm/`**`extractors.xml`**.

Parameter	Description	Example setting
`<Signature>`	Format signature, which is a 'standard' sequence of values (bytes) with initial displacement and masks. This sequence is required for unambiguous definition of file format by content of files. The signature can include up to 16 triades of the following format: `[OFFSET]:BYTE_VALUE/[MASK]` String signatures like "BZh" for BZIP2 are also supported. String signature means specified byte sequence from 0 displacement. The search of string signatures is not case-sensitive. Optional parameters: • `OFFSET` To specify displacement, use '`?`' symbol. This will enable searching for a pattern in first N bytes of a file. For example, to search the pattern in first 500	Example for a WAV-heading signature: `Sign = "'RIFF', 0x8: 'WAVEfmt'"` Starting from 0 displacement, there is a sequence '`R`','`I`','`F`','`F`', then from 8 displacement there should be '`W`','`A`','`V`','`E`','`f`','`m`','`t`'

	bytes, specify '`500?`' displacement. • `MASK`	
`<FilenameCharsets>`	In case file name encoding differs from the standard one (ANSI, UTF), the System processes filename as corresponding to encodings listed in this setting. To list several values, separate them with comma; the System sequentially tries to process the filename in listed encodings while it finds a matched one.	`UTF-8`, `cp1251`, `cp866`
`<SpeedInKBs>`	Proposed extractor operation speed per 1 GHz of the processor (in Kb/s). Can take integer values starting with 1.	`100`
`<UseLog>`	Extractor logging	• `true` • `false`
`<TimeoutInSec>`	Waiting time till file processing is completed (in seconds)	`1000`
`<MinFileSizeInBytes>`	Minimum file size (in bytes) subjected to unpacking. This limit enables the System performance increase due to skipping small files unpacking. **Note:** Processing of files under 10 bytes is not supported. This limitation also applies to *Clipboard* events.	`1024`
`<MaxTextPlainSizeInKb>`	Maximum size of a file whose signature is not defined. If the threshold value is exceeded, the file is automatically considered binary and is not sent to `iw_cas`.	`25600`
`<PreCommand>`	Command to be executed before or after data extraction	`bin/dwg-extractor -i ${SRC} -t ${OUTDIR}/out.txt -u ${OUTDIR}/out.vectxt`
`<PostCommand>`		

Note:

Some JPEG files, intercepted by the System, can be spliced with archives potentially containing sensitive data. By default, the System does not extract text from such files. In order to configure text extraction for spliced JPEG files:

1. Open `/opt/iw/tm5/etc/config-perm/bookworm/extractors_customization.xml` file for editing:
2. Inside `<extractors>` element, paste the element with the following contents:

```
<extractor profile_name="jpg" key="dY5k14hWa0nivGOvSP4BUe0vTJdME5IY4xjNsqKl">
        <PostCommand>bin/7zz x -y -p{} ${SRC} -o${OUTDIR}</PostCommand>
</extractor>
```

3. Save your changes and close the file.
4. Restart Traffic Monitor services:

```
iwtm restart
```

9.4 Configuring Settings of Search Using Wildcards

When creating a query in the "**Events**" section of Traffic Monitor Management Console, you can specify senders and recipients. To specify values of their contacts or contacts metadata, you can use wildcards at the end of the string. For search using wildcards, the System considers contacts and their metadata as keywords and indexes all possible prefixes of keywords. Prefix is an initial part of a word. If System searches for a certain prefix in an index, then it finds all keywords containing the prefix.

The System perfoms indexing starting with the minimum prefix length that is determined by the parameter `min_prefix_len` in the `is.conf` file. By default, the `min_prefix_len` value is 3.

Example:

There are users having *ex_user*, *exact_user* и *example_user* logins. The `min_prefix_len` value is 3.

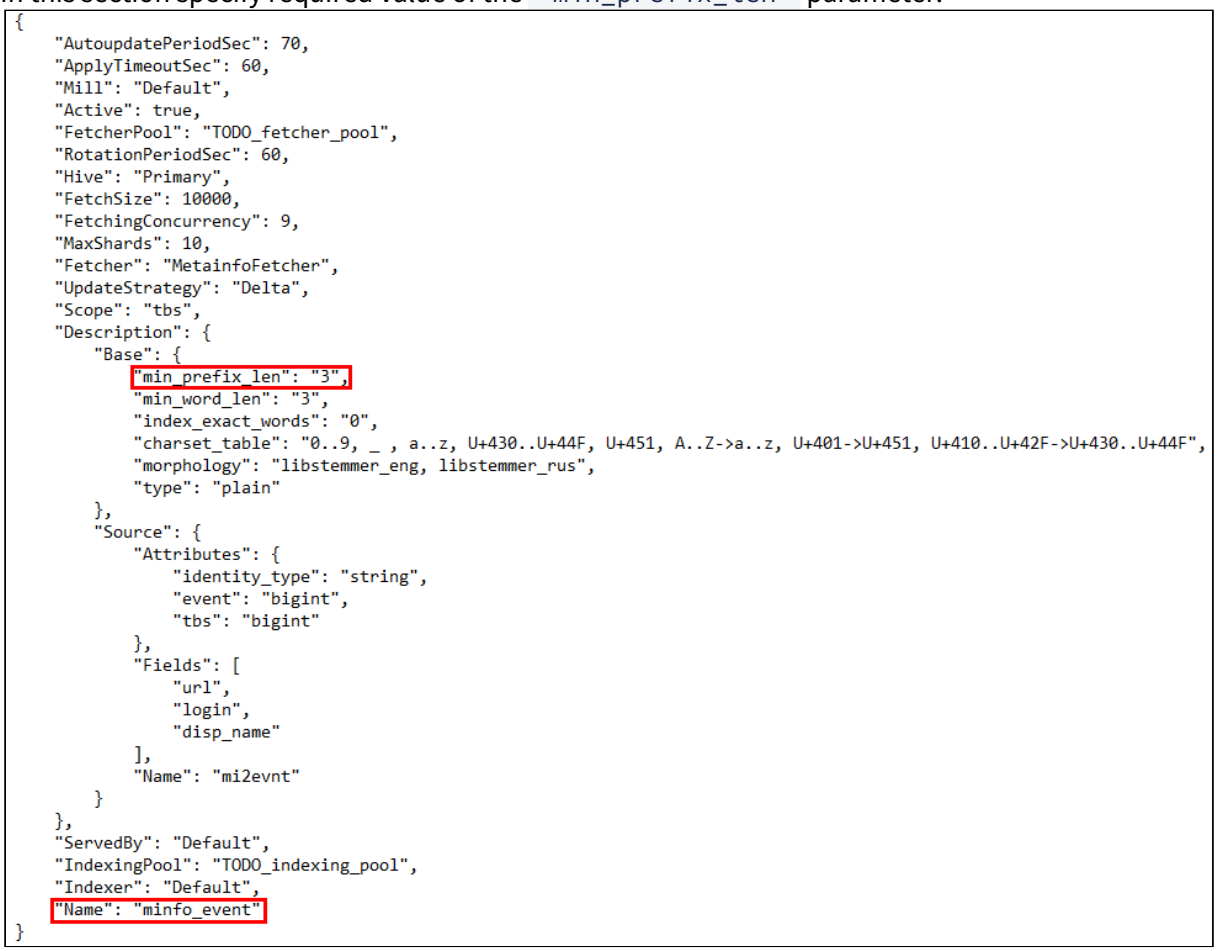
If you create a query with the `"exa*"` mask, then as a result the events with *exact_user* and *example_user* are to be found. At the same time a query with the `"ex*"` mask will not return any result.

If you change the `min_prefix_len` value to 2, then the result of the search for `"ex*"` mask will contain events with all three users – *ex_user*, *exact_user* and *example_user*.

To change the prefix minimum length:

1. On Traffic Monitor server, go to the `/opt/iw/tm5/etc/` directory.
2. Open the `is.conf` file.
3. In the `"Indices"` root field, find the section with the `"Name": "minfo_event"` parameter.

4. In this section specify required value of the `"min_prefix_len"` parameter.



```
{
    "AutoupdatePeriodSec": 70,
    "ApplyTimeoutSec": 60,
    "Mill": "Default",
    "Active": true,
    "FetcherPool": "TODO_fetcher_pool",
    "RotationPeriodSec": 60,
    "Hive": "Primary",
    "FetchSize": 10000,
    "FetchingConcurrency": 9,
    "MaxShards": 10,
    "Fetcher": "MetainfoFetcher",
    "UpdateStrategy": "Delta",
    "Scope": "tbs",
    "Description": {
        "Base": {
            "min_prefix_len": "3",
            "min_word_len": "3",
            "index_exact_words": "0",
            "charset_table": "0..9, _ , a..z, U+430..U+44F, U+451, A..Z->a..z, U+401->U+451, U+410..U+42F->U+430..U+44F",
            "morphology": "libstemmer_eng, libstemmer_rus",
            "type": "plain"
        },
        "Source": {
            "Attributes": {
                "identity_type": "string",
                "event": "bigint",
                "tbs": "bigint"
            },
            "Fields": [
                "url",
                "login",
                "disp_name"
            ],
            "Name": "mi2evnt"
        }
    },
    "ServedBy": "Default",
    "IndexingPool": "TODO_indexing_pool",
    "Indexer": "Default",
    "Name": "minfo_event"
}
```

5. Save the file.
6. Perform the re-indexing. For this:
    a. Stop the **iw_is** process.
    b. Delete the `/var/lib/sphinx/indices/minfo_event` directory.
    c. Start the **iw_is** process.
    The re-indexing may take some time depending on the number of events in Traffic Monitor. During this time, if you perform the `iwtm status iw_is` command, you will get the query execution error.

9.5 Configuring Integration with Data Analysis Service

In Traffic Monitor, you can use the *Search for analogues* feature for events that contain attachments (for details, see *InfoWatch Traffic Monitor. User Guide. Search for Analogues*).

To enable the *Search for Analogues* feature, configure integration between Traffic Monitor and Data Analysis Service:

1. Install Data Analysis Service. For details, see *InfoWatch Data Analysis Service. Installation, Configuration and Administration Guide. Installing Data Analysis Service*.
2. Set up a connection with Data Analysis Service using the **platform_conf_ctl** utility. For this:
    a. On the Traffic Monitor server, open the `/opt/iw/tm5/bin/` directory where the utility is located.

b. Run the command:
```
./platform_conf_ctl --das-set-uri <host_DAS>:30050
                    --das-set-token <token_value>
                    --das-set-enabled
                  1
```
where `<host_DAS>` is the hostname or IP address of the Data Analysis Service server, `<token_value>` is the value of the token for connection with Data Analysis Service. By default, use the `2exwo92iwpczl1hy6jxy` token value.
You can find the actual value of the token:
- On the central node of the cluster, in which the Data Analysis Service server is located, via the command line:
  i. Go to the `<distr>/platform` directory. `<distr>` is the directory, where the distribution kit of Data Analysis Service or any platform product was extracted;
  ii. Run the command:
  `./setup.py tokens list`
  As a result, you will see a list of tokens for connecting to the Platform. The actual Data Analysis Service authorization token will be displayed opposite the *DMServer* identity.
- In the Platform product management console, in the **Token for Device Monitor Server** field of the **Global settings** section.
  Available if the Platform product with web interface is installed in the cluster and the web interface contains such a setting.

3. In the `"tmhost": "set_tmhost_here"` line of the `/opt/iw/tm5/etc/attributor.conf` configuration file, specify the IP address of the Traffic Monitor server instead of `set_tmhost_here` . Save the changes.

> **Note:**
>
> Here you can also change:
>
> - the logging level for the **iw_ attributor** process. For this, in the " `Logger` " section, set the required value for the " `level` " parameter. For details, see *Logging System Work*.
> - minimum size of contents sent to Data Analysis Service for processing. For details, see *below*.
> - maximum number of contents sent in a single request to Data Analysis Service for processing. For details, see *below*.

4. Get a license for Data Analysis Service from InfoWatch engineers and upload it to the Traffic Monitor Management Console.
5. Enable file queue in the database. For this, connect to the DB server (for details, see *Database Administration*) and run the following command:
```
UPDATE object_queue
SET enabled = 1
WHERE object_queue_id = 5;
```
6. Unmask the **iw_ attributor** process. For this, on the Traffic Monitor server, run the command:
```
systemctl unmask iw_attributor
```

7. Run the **iw_ attributor** process:
   `iwtm start iw_attributor`
8. Make sure the **iw_ attributor** process is running and have the *active* status.
   ```
   Service iw_attributor.service is active (running); enabled state: loaded (enabled)
   ```
   For this, run the command:
   `iwtm status iw_attributor`
9. Enable autorun for the **iw_ attributor** process:
   `iwtm enable iw_attributor`

> **Note:**
>
> Data Analysis Service starts processing events since you have configured this setting. For events that happend before, the *Search for analogues* feature does not work and those events will not be taken into account if you run *Search for analogues* for other events.

To disable the *Search for Analogues* feature:

1. Stop the **iw_ attributor** process. For this, on the Traffic Monitor server, run the command:
   `iwtm stop iw_attributor`
2. Disable autorun for the **iw_ attributor** process:
   `iwtm disable iw_attributor`
3. Disable file queue in the database. For this, connect to the DB server and run the following command:
   ```
   UPDATE object_queue
   SET enabled = 0
   WHERE object_queue_id = 5;
   ```

Setting a minimum size of contents sent to Data Analysis Service for processing

To filter a flow of contents sent to Data Analysis Service for processing, you can set their minimum size for sending. It allows you to filter out uninformative contents of a small size, e.g. texts extracted from images and containing less than 50 symbols. To set a minimum size:

1. On the Traffic Monitor server, open the `/opt/iw/tm5/etc/attributor.conf` configuration file for editing.
2. In the `"TextSizeMinimumInBytes"` field, specify the minimum size in bytes.
3. Save the changes.

By default, the `"TextSizeMinimumInBytes"` field has the value of 500 bytes.

> **Note:**
>
> The parameter limits sending data only by text content size. If the attachment contains an image or a container of another kind, only the size of extracted text will be considered.
>
> If the attachment is an archive, then each element of its content will be filtered separately.

Setting a number of contents sent to Data Analysis Service in a request

If too much contents are sent to Data Analysis Service in a single query, the time during which Traffic Monitor waits for a response from Service can be exceeded. In this case, the request may not be processed. To prevent this, the maximum allowable number of contents in a request is limited by the `contentAmountPerRequest` parameter.

Traffic Monitor combines groups of contents extracted from different files into one request if the total number of these contents does not exceed the parameter value.
If the number of contents from a single file exceeds the `contentAmountPerRequest` parameter value, then nevertheless they will be sent in a single request.

To change the maximum allowable number of contents sent in a single request:

1. On the Traffic Monitor server, open the `/opt/iw/tm5/etc/attributor.conf` configuration file for editing.
2. In the `"contentAmountPerRequest"` field, specify the number of contents.
3. Save the changes.

By default, the `"contentAmountPerRequest"` field has the value of 1000.

9.6 Configuring Connection with Data Storage

To optimize data storing and increase DBMS productivity, you can install external storage of binary data – Data Storage. For details, see *InfoWatch Traffic Monitor. Installation Guide. System Deployment Schemes and Installation Types* and *Installing Data Storage*.

To configure the connection between Traffic Monitor and Data Storage:

1. Set the `debug` logging level for the `iw_x2db` process. This will allow you to make sure the connection is successful when it is configured. For details, see *Logging System Work*.
2. On the Traffic Monitor server, run the command:
`platform_conf_ctl --ds-set-enabled <bool> --ds-set-uri <platform-uri> --ds-set-token <token>`
where you specify the following parameters:

Parameter	Description	Examples
`--ds-set-enabled <bool>`	Using Data Storage. To store binary data of events in Data Storage, specify the "1" value. If you specify the "0" value, the System will not use Data Storage for storing binary data of new events	`--ds-set-enabled 1`
`--ds-set-uri <platform-uri>`	Hostname or IP address of the Data Storage server. The `http` protocol and the `40731` port are used by default.	`--ds-set-uri my.platform.com` `--ds-set-uri https://my.platform.com:40731`

Parameter	Description	Examples
`--ds-set-token <token>`	Data Storage authorization token The default value of the token: `1exwo92iwpczl1hy6jxy` **Warning!** To improve security, it is not recommended to use the default value of the token. You can find the actual value of the token: • On the Data Storage server, via the command line. For this: a. Go to the directory where the Platform distribution kit was extracted. b. Run the command: `./setup.py tokens list` As a result, you will see a list of tokens for connecting to the Platform. The actual Data Storage authorization token will be displayed opposite the *DataStorageUser* identity. You can refresh the Data Storage authorization token by running the command: `./setup.py tokens refresh DataStorageUser` • In the Platform product management console, in the **Token for Data Storage User** field of the **Global settings** section. Available if the Platform product with web interface is installed on the server with Data Storage and the web interface contains such a setting. In this case, you can refresh the token by pressing the button  .	`--ds-set-token default` `--ds-set-token 1exwo92iwpczl1hy6jxy`

If the connection is succesfull, a message of the following type will appear in the `/var/log/infowatch/x2db.log` log file:

```
[DEBUG   ] : <DataStorage> platform request headers: GET /v2/logon?extAPIToken=1exwo92iwpczl1hy6jxy HTTP/1.1
Host: my.platform.com
User-Agent: tmOperator
Accept: application/json
X-IW-ACTIVEPRODUCT: TM

[DEBUG   ] : <DataStorage> platform response: HTTP/1.1 200 OK
```

After configuring is complete, you can set the logging level for the `iw_x2db` process that was set before configuring.

To view current settings of the connection with Data Storage:

On the Traffic Monitor server, run the command:
`platform_conf_ctl`

To break the connection with Data Storage:

On the Traffic Monitor server, run the command:
`platform_conf_ctl --ds-remove`
Data in the storage will no longer be available.

If you want to stop storing binary data of events in Data Storage but have access to already stored data, run the command:
`platform_conf_ctl --ds-set-enabled 0`
New events will be stored entirely in Traffic Monitor DB, starting with the next daily tablespace, i.e. starting tomorrow in UTC+0.

9.7 Logging System Work

For the convenience of monitoring processes, the logging subsystem has 6 levels:

Level	Description
`fatal`	Writes errors that are critical for the process operation
`error`	Writes messages about errors that are not critical for the process operation
`warning`	Writes information about potentially dangerous cases
`Information`	General information about the process that can be useful (process start/stop, applying configurations, etc.)
`debug`	Writes information which is mostly used to diagnose the service operation (IT, system administrators and so on)
`trace`	Mostly used by programmers to track the code

Configuring logging levels

> **Warning!**
>
> Please, note that choosing a more detailed logging level may significantly reduce the System performance.

Configuration files are located in the `/opt/iw/tm5/etc` directory. By default, the `warning` level is used for all processes.

To change the logging level for Traffic Monitor Server processes:

1. In the configuration file of the required process, find the `Logging` section and set the needed value for `GlobalLevel` parameter.

2. Save changes.
3. After you have changed the system log, restart the process:
   `iwtm restart <service_name>`

By default, all process logs are stored in the `/var/log/infowatch` directory.

For extractors which do not have a separate configuration file, the logging level can be set in the **/opt/iw/tm5/etc/config-perm/bookworm/extractors.xml** configuration file:

1. Open the configuration file.
2. For `text-extractor` parameter, add the value `-l <level>,` where `<level>` is the logging level. For example:
   `< Command> bin/text-extractor -l error -t ooxml -i ${OUTDIR}" < Command>`
3. After you have made changes, restart processes:
   `iwtm restart bookworm`
   `iwtm restart warpd`

> **Note:**
>
> This type of configuring the logging level is only available for extractors of the following formats: docx, html, msoffice_xml, msole, odp, odt, pptx, xlsx, xml.

9.8 File Queues

To enable loading the existing object queues:

1. Stop the following services: `iw_icap`, `iw_deliver`, `iw_luaengined`, `iw_is`, `iw_smtpd`, `iw_xapi_xapi`, `iw_xapi_puppy`, `iw_analysis`, `iw_messed`, `iw_proxy_http`, `iw_proxy_icq`, `iw_proxy_smtp`, `iw_capstack`.
2. Place objects from the error queue to the required queue.
3. Wait until all objects in the file queue are processed.
4. Run the following services: `iw_icap`, `iw_deliver`, `iw_luaengined`, `iw_is`, `iw_smtpd`, `iw_xapi_xapi`, `iw_xapi_puppy`, `iw_analysis`, `iw_messed`, `iw_proxy_http`, `iw_proxy_icq`, `iw_proxy_smtp`, `iw_capstack`.

To start all the services, run the command:

```
iwtm start
```

To check if the processes are started, use the following command:

```
iwtm status
```

Commands for starting/stopping separate services:

Starting	Stopping
`iwtm start icap`	`iwtm stop icap`

Starting	Stopping
`iwtm start deliver`	`iwtm stop deliver`
`iwtm start luaengined`	`iwtm stop luaengined`
`iwtm start is`	`iwtm stop is`
`iwtm start smtpd`	`iwtm stop smtpd`
`iwtm start xapi_xapi`	`iwtm stop xapi_xapi`
`iwtm start xapi_puppy`	`iwtm stop xapi_puppy`
`iwtm start analysis`	`iwtm stop analysis`
`iwtm start messed`	`iwtm stop messed`
`iwtm start proxy_http`	`iwtm stop proxy_http`
`iwtm start proxy_icq`	`iwtm stop proxy_icq`
`iwtm start proxy_smtp`	`iwtm stop proxy_smtp`
`iwtm start capstack`	`iwtm stop capstack`

Objects in the System are moved by means of file queues:

Path to the directory of the queue	Format of files in the queue
queue/analysis	`.xml & .dat`
`queue/smtp`	`.xml & .dat`
`queue/db`	`.xml & .dat`
`queue/blackboard`	`.dat`
`queue/blackboard_errors`	`.dat`
`queue/is`	`.xml & .dat`
`queue/x2x`	`.xml & .dat`
`queue/errors`	`.xml & .dat`

Path to the directory of the queue	Format of files in the queue
`queue/x2x-errors`	`.xml & .dat`
`queue/x2db-errors`	`.xml & .dat`
`queue/final-errors`	`.xml & .dat`

Each queue contains additional technical queues which allow you to monitor for processing objects: `.db, .in, .out`.

Note:

For the `queue/smtp` process, the processing can be divided into the following steps:

1. `queue/smtp/.in` – as the **iw_smtpd** service receives objects from Postfix, a file is generated in this queue. If for some reason, the object processing in this queue is delayed, please, check **iw_smtpd** or **iw_proxy_smtp** service.
2. `queue/smtp/.db` – when the processing is complete, the file is moved from the `.in` queue to the `.db` queue. EML mail objects from **iw_xapi** are placed in this queue.
3. `queue/smtp/.out` – the object is placed in this queue by the **iw_messed** service.

Note that:

Queues of the **iw_is** service are additionally divided into:

- `queue/is/cmd/` – a file queue of **iw_is** commands;
- `queue/is/fetching/` – a queue of commands for loading;
- `queue/is/indexing/` – a queue of commands for indexing;
- `queue/is/errors/` – an error queue.

Queue name	Services that place an event in the queue	Services that remove an event from the queue
`queue/analysis`	`iw_xapi_xapi`, `iw_xapi_puppy`	`iw_analysis`
`queue/smtp`	`iw_xapi_xapi`, `iw_xapi_puppy`, `iw_smtpd`, `iw_proxy_smtp`, `iw_capstack`	`iw_messed`
`queue/db`	`iw_messed`, `iw_analysis`, `iw_icap`, `iw_proxy_icq`, `iw_proxy_http`	`iw_x2x`
`queue/blackboard`	`iw_x2db`, `iw_deliver`	`iw_blackboard`
`queue/blackboard_errors`	an error when processing `iw_blackboard`	-

Queue name	Services that place an event in the queue	Services that remove an event from the queue
`queue/x2x`	`iw_x2x`	`iw_x2db`
`queue/is`	`iw_metainfo_fetcher`	`iw_is`
`queue/errors`	all services in case of processing errors	`iw_rammer`
`queue/x2x-errors`	errors when processing `iw_x2x`	
`queue/x2db-errors`	errors when processing `iw_x2db`	
`queue/final-errors`	`iw_rammer`	-

You can enable collecting statistics on file queues in the service configuration file which is located in the `/opt/iw/tm5/etc` directory. For this, in the *Statistic* section, set the **Enabled** parameter value to `true`.

9.9 Recovering the System after Failures

You can recover the System when server failures occur or during a network outage by completing the steps described below:

1. Stop Traffic Monitor Server processes by running the command:
   `iwtm stop`
2. Move objects from the `queue/x2db-errors` error queue to the `queue/x2x` queue. For this, use **iw_qtool** utility (see the Knowledge Base article *Moving of Objects between Queues (iw_qtool Utility)*).
3. If Traffic Monitor Server works with installed PostgreSQL DBMS, then, to check the connection, run the following command:
   `su - iwtm`
   `psql -p 5433 -h <DB_server_IP_address> postgres iwtm`
   `psql -p 5433 -h 127.0.0.1 postgres iwtm` - to connect local database
   If the check is successful, then, in response to the execution of the command, PSQL invitation will be displayed:
   `postgres=#`
   If Traffic Monitor Server works with installed Oracle DBMS, then, to check the connection, run the following command:
   `sqlplus db_login/db_password@tns_name`
   where `db_login` and `db_password` are the name and the password of the DB schema owner respectively and `tns_name` is the name of TNS service.
   If the check is successful, then, in response to the execution of the command, SQL *Plus invitation will be displayed:
   `SQL>`
4. Run the Traffic Monitor Server processes by executing the command:
   `iwtm start`

If after Traffic Monitor launch the System is not recovered:

1. Collect the diagnostic information, i.e. logs.
   For details on logging, see *Oracle Database Logging*, *PostgreSQL Database Logging*, *Logging System Work*.
2. Check the system log for errors. The path to the system log: `/var/log/messages` .
3. Analyze the collected information. You can find typical errors and methods of their solution in the Knowledge Base article: *Errors in PostgreSQL database*. If needed, consult the DB administrator or technical support service.

9.10 Managing Languages Including Morphological Support

Managing languages including morphological support in Traffic Monitor is being performed through configuration file change and installation of package with language dictionary.

Depending on settings specified during installation, two and more languages can be installed in the System (find information on installation in the document *InfoWatch Traffic Monitor. Installation Guide*). Russian and English morphology dictionaries are preinstalled (even if it was not specified during installation).

Upon installation, user also can:

- Adding New Language for Events Search. Morphology and Adding Terms;
- Updating Installed Language Including Morphological Support;
- Deleting Search Language and Terms.

9.10.1 Adding New Language for Events Search. Morphology and Adding Terms.

To add a new language to the System for event search:

1. Stop thy **iw_indexer** and **iw_is** processes:
   ```
   iwtm stop indexer
   iwtm stop is
   ```
2. Go to the local repository that keeps packages downloaded during the installation:
   ```
   cd /opt/iw/distr
   ```
3. Go to directory that matches current Traffic Monitor version.
4. Select the required package and install it using the command:
   ```
   rpm -i <package name>
   ```
   For example, to install French dictionary package, enter the command:
   ```
   rpm -i iwtm-sphinx-dict-fra-7.0.0-792.x86-64.rpm
   ```
5. Run the processes:
   ```
   iwtm start indexer
   iwtm start is
   ```

   > **Note:**
   >
   > Installing a new dictionary causes full database re-indexing, which may take some time.

   To enable morphology for events search:
6. Stop the **iw_indexer** and **iw_is** processes:
   ```
   iwtm stop indexer
   iwtm stop is
   ```

7. Open the **sphinx_options.ini** file using the command:
`mcedit /opt/iw/tm5/etc/sphinx_options.ini` and add the required language key to `sphinx_language` parameter.
For example, to activate the morphology for Russian and Turkish languages, specify the following parameter values:
```
sphinx_languages = 'rus tur'
```

> **Note:**
>
> To set the language priority, put the language parameter to be the last in the list of `sphinx_languages` parameters of **sphinx_options.ini** file.

8. When using PostgreSQL DBMS, open the **database.conf** file to add terms:
`mcedit opt/iw/tm5/csw/postgres/database.conf` When using Oracle DBMS, open the **database.conf** file to add terms:
`mcedit /opt/iw/tm5/csw/oracle/database.conf`
9. Add the language key with morphology support from `cfdb_language` parameter, for example,
```
\set cfdb_language 'rus'
```

> **Note:**
>
> The full list of language keys with morphology support can be found in *List of languages including morphological support*.

10. Run the **iw_indexer** and **iw_is** processes:
`iwtm start indexer`
`iwtm start is`

9.10.2 Updating Installed Language Including Morphological Support

To update existing dictionary:

1. Go to directory with dictionary packages:
`cd /opt/iw/disrt/7.0.0/`
2. Run package installation:
`rpm -i <package name>`
For example:
`rpm -i iwtm-sphinx_dict-deu-7.0.0.792.x86_64.rpm`

> **Note:**
>
> Updating dictionary, database reindexing will not take place. Language support will be performed only for new events.

9.10.3 Deleting Search Language and Terms

To delete a dictionary from the System:

1. Stop the **iw_indexer** and **iw_is** processes:
   `iwtm stop indexer`
   `iwtm stop is`
2. Open **database.conf** file if you use PostgreSQL DBMS:
   `mcedit /opt/iw/tm5/csw/postgresql/database.conf`
   If you use Oracle DBMS, open file **database.conf** to add terms:
   `mcedit /opt/iw/tm5/csw/oracle/database.conf`
3. Delete code of language with morphology support from `define cfdb_language` parameter.
4. Delete indexing files from `/var/lib/sphinx/` directory.
5. Start processes:
   `iwtm start indexer`
   `iwtm stop is`

After you delete a dictionary, the database is completely reindexed.

> **Note:**
>
> If you want to save the language in Categories and Terms, then add the code of the language with morphology support to the `cfdb_language` parameter of the **database.conf** configuration file just before updating the System. This will also restore the language with morphology support. If you want to delete the language with morphology support after the System update, repeat the steps described above.

9.11 Configuring Forwarding Data to SIEM

Traffic Monitor can be integrated with SIEM systems (Kaspersky Unified Monitoring and Analysis and others). Integration means forwarding consolidated information from all installed TM components to SIEM.

Configuring this functionality is possible using the script **/opt/iw/tm5/bin/config/iwtm-siem.conf.py**, which is available on Traffic Monitor server.

Information from TM is available for SIEM system:

1. By means of a table view:
   - Events, registered in TM;
   - TM console user session audit;
2. By means of rsyslog:
   - Logging in and out TM Linux server;
   - Events, registered in TM Linux server system log, including information on logging in and out Oracle or PostgreSQL DB users;
   - State of services on TM Linux server or group of servers.

In this chapter:

- Configuring SIEM Side;
- Configuring TM Side;
- Type of Logs, Transmitted to SIEM.

9.11.1 Configuring SIEM Side

To prepare SIEM to recieving data from TM:

1. In SIEM tables to extract information from:

- `IWTM.ARC_VIEW_OBJECTS`
  TM events
- `IWTM.ARC_VIEW_AUDIT_LOG`
  TM user audit

- Create SIEM account to access DB tables.
- Configure processing data, which has been extracted from TM DB.
  To analyse table views of TM events and user audit see articles *Table Representation of TM Events* and *User Audit Table View*.

> **Warning!**
>
> While configuring integration, additional modules from SIEM system vendor, which allow SIEM system to work with table views, can be required. For example, HP Flex Conector module is required for integration with HP ArcSight.

Table Representation of TM Events

Connect to TM database using the created **siem** user (see *Creating Siem User*). The `IWTM.ARC_VIEW_OBJECTS2` table contains the desired data.

> **Note:**
>
> If for importing data to SIEM, you need to filter the data, you can use `capture_date` or `insert_date` fields for filtering. We recommend filtering data by the `insert_date` field. Filtering data by the `object_id` field is not supported.

Attribute name	Data type	Description	Possible values	Example
`object_id`	number(20)	*Event ID* attribute of the event	Event ID in TM database. Is always present.	110
`monitorcode`	varchar2(4000)	*Event type* attribute of the event	The event type determines how the data is transferred. Possible values: • External device • Printing	Email

Attribute name	Data type	Description	Possible values	Example
			• ICQ • Skype • XMPP • Mail.Ru Agent • Telegram • VK (ВКонтакте) • Facebook • WhatsApp • MS Lync • FTP • Email • Web email • Web message • Clipboard • Cloud storage • Keyboard input • Terminal session • Network resources • File reading • File writing	
`protocol`		*Protocol* attribute of the event	Possible values: • OSCAR • Skype • XMPP • MMP • SIP • FTP • POP3 • IMAP • MAPI • SMTP • HTTP • HTTPS • NRPC • WhatsApp • None	
`verdict`	varchar2(12)	*Verdict* attribute of the event	Possible values: • Quarantined • Forbidden • Allowed	Forbidden
`date_of_capture`	varchar2(50)	*Date of capture* attribute in ISO 8601 with specifying the		2014-06-19T18:56:26+04:00

Attribute name	Data type	Description	Possible values	Example
		timezone of the data capture		
`capture_data`	timestamp	Event interception date (in UTC, without specifying the timezone) in `timestamp` format. In PostgreSQL DBMS, the attribute has an increased accuracy to milliseconds. Is used for fast data filtering.		timestamp
`insert_date`	timestamp	Event insertion date (in UTC, without specifying the timezone) in `timestamp` format		timestamp
`device`	varchar2(256)	*Device name* attribute of the event	Name of the device, to which or from which data has been copied. Or a name of the printer, to which a document from the intercepted event has been sent. Can be blank.	Kingston USB Drive 5.0
`Web-source`	varchar2(256)	*Resource* attribute of the event	Address of the visited web resource or address of cloud storage. Can be blank.	Yahoo.com
`recipients contacts`	clob	*Recipients* attribute of the event	List of recipients separated by commas in `<contact type>:<contact value>` format. Can be blank.	email:ivanov@infowatch.ru, email:petr.petrov@infowatch.ru
`recipients fullname`	clob	*Recipients* attribute of the event	List of recipients' full names separated by commas.	Ivanov Ivan Ivanovich,

Attribute name	Data type	Description	Possible values	Example
			The order is the same as for `recipients`. Names are specified only if recipients have been identified. Can be blank	Petrov Petr Petrovich
`recipientsdomainaccountname`	clob	*Recipients* attribute of the event	List of identification keys of the `auth` type for all recipients. Keys in the list are separated by commas. The order is the same as for `recipients`. Can be blank.	Ivanov@iw, Petrov@iw
`senderdomainaccountname`	clob	*Senders* attribute of the event	Identification key of `auth` type for the sender. Can be blank.	Petrov@iw
`senderfullname`	clob	*Senders* attribute of the event	Sender's full name. Can be blank.	Petrov Petr Petrovich
`sendercontacts`	clob	*Senders* attribute of the event	Sender's contact in `<contact type>:<contact value>` format. Can be blank.	email:petr.petrov@infowatch.ru
`sendermachinedomainname`	clob	*Workstation* attribute of the event	Domain name `dnshostname` of the sender's workstation. Can be blank.	XP-PETROV
`sendermachineip`	clob	*Workstation* attribute of the event	IP address of the sender's workstation. Can be blank.	192.0.2.0
`perimetersin`	clob	Name of the perimeter the event has entered	List of names of the perimeters which the event has entered.	Company perimeter

Attribute name	Data type	Description	Possible values	Example
			Names in the list are separated by commas. Can be blank.	
`perimetersout`	clob	Name of the perimeter which the event has left	List of names of the perimeters which the event has left. Names in the list are separated by commas. Can be blank.	Company perimeter
`tags`	clob	*Tag* attribute of the event	List of tags, separated by commas. Can be blank.	New
`categories`	clob	*Category* attribute of the event	List of categories, separated by commas. Can be blank.	Confidentially
`text_objects`	clob	*Text object* attribute of the event	List of text objects detected in the intercepted event. Text objects in the list are separated by commas. Can be blank	special control
`fingerprints`	clob	*Form*, *Sample document*, *Stamp*, *DB unloading* attributes of the event	List of forms, sample documents, stamps, and DB unloadings detected in the intercepted object. Elements in the list are separated by commas. Can be blank.	Application form.doc
`protecteddocuments`	clob	*Protected object* attribute of the event	List of protected objects detected in the intercepted object. Elements in the list are separated by commas. Can be blank.	Lease contract
`policies`	clob	*Policy* attribute of the event	List of policies triggered by the intercepted object. Policy names are separated by commas.	New employees monitoring policy

Attribute name	Data type	Description	Possible values	Example
			Can be blank.	
`violationtype`	varchar2(40)	*Rules group* attribute of the event	Possible values: • *Copy* (*Copying violation*) • *Placement* (*Placing violation*) • *Transfer* (*Transferring violation*) Can be blank.	Copy
`violation_level`	List	*Violation level* attribute of the event	Possible values: • *High* • *Medium* • *Low* • *No violation* Can be blank.	Medium
`userdecision`	varchar2(27)	User decision attribute of the event	Decision of the Console user. Possible values: • *Violation* • *NoViolation* • *NotProcessed* • *AdditionalProcessingNeeded* Can be blank	Violation
`filepath`	clob	*File path* attribute of the event	List of paths to files. Paths in the list are separated by commas. Can be blank.	\\xp-petrov\C$ \example.txt
`attachments`	clob	*Attachment name* attribute of the event	Names of attachments separated by commas	example.txt, Petrov.doc, 3.jpg
`url`	varchar2(4000)	*URL* attribute of the event	Full address from which data has been sent Can be blank.	192.0.2.0/3.jpg

Attribute name	Data type	Description	Possible values	Example
`capture_server_ip`	varchar2(256)	*IP of interception server* attribute of the event	IP address of the server, on which data has been intercepted. Can be blank.	10.60.21.34
`capture_server_hostname`	varchar2(256)	*Name of interception server* attribute of the event	Name of the server, on which data has been intercepted. Can be blank.	iwtm.infowatch.ru

Warning!

The number of elements to display within one cell of one record is limited to 1000 (For example: 1000 recipients or extracted files in the event).

User Audit Table View

Connect to TM DB using created **siem** user (see *Creating Siem User*). `IWTM.ARC_VIEW_AUDIT_LOG` contains desired data.

Attribute	Data type	Description	Example
`audit_log_id`	Number(20)	TM user session audit entry ID.	12345
`change_date`	Varchar2(50)	Date and time of registered user action	05.08.2015 09:35:10.641446000
`user_login`	varchar2(256)	Login of user, who performed an action	Admin
`user_fullname`	varchar2(256)	Full name of user, who performed an action	Petrov Petr Petrovich
`user_email`	varchar2(256)	Email of user, who performed an action	petr.petrov@infowatch.ru
`operation`	varchar2(40)	Type of action, performed by user Possible values: • restart (restart) • delete_hash (hash deletion)	Edit

Attribute	Data type	Description	Example
		• sync (synchronization) • add_tag (adding tag) • remove_tag (removing tag) • run (performing query) • start (starting) • stop (stopping) • view (viewing) • create (creation) • update (editing) • delete (deletion) • login_failure (unsuccessful login attempt) • login (successful login) • logout (logout) • change_password (password change) • decision_update (changing user decision) • remove_tag (changing event tag) • delete_ref (deleting reference) • copy (copying) • move (move) • commit (applying changes in system configuration) • rollback (cancellation of changes in system configuration) • draft (saving changes in system configuration) • add (adding) • import (import) • export (export)	
`entity_type`	varchar2(40)	Type of object on which the action was carried out. Possible values: • AgentJob (diagnostic data) • Adlibitum (Adlibitum) • Agent (service) • Classifier (classifier)	Dashboard

Attribute	Data type	Description	Example
		• NetworkSettings (network parameters) • NotificationSettings (system status) • ObjectReport (events unloading) • Query (launching objects search) • QueryReportRun (report creation) • Setting (settings) • UpdateSystem (system update) • Category (category) • Dashboard (dashboard) • DashboardWidget (widget) • EtForm (sample form) • EtStamp (sample stamp) • EtTable (sample table) • Fingerprint (sample documents) • LdapContact (LDAP contact) • LdapGroup (LDAP group) • LdapPerson (LDAP person) • LdapStatus (LDAP person status) • LdapWorkstation (LDAP workstation) • Perimeter (perimeter) • Policy (policy) • ProtectedDocuments (protected object) • Report (report) • Role (role) • Selection (selection) • ServiceLog (service log) • SystemList (list of resource topics) • SystemListItem (list of resources with definite topic) • Tag (tag) • Term (term) • TextObject (text pbject) • User (user)	

Attribute	Data type	Description	Example
		• VisibilityArea (visibility scope) • Config (configuration) • License (license) • Object (object) • ProtectedCatalog (protected object catalog)	
`entity_display_name`	varchar2(4000)	Name of object, on which the action was carried out.	Statistics1
`property_changes`	clob	Description of actions in json format. Filled only in case of managing users, roles, areas and logging in and out of the management console. Three different formats are possible.	
	-	**Format №1.** Relevant for events of role and visibility scope management.	
		`{ “old”: { “<OBJECT TYPE>”:{ { “<FIELD>”: “<VALUE>”, “<FIELD>”: “<VALUE>” } } } { “new”: { “<OBJECT TYPE>”:{ { “<FIELD>”: “<VALUE>”, “<FIELD>”: “<VALUE>” } } }`	`{ "old":{ "visibilityareas":{ } }, "new":{ "visibilityareas":[ { "VISIBILITY_AREA_ID":"F00207A1E7E7743EE0433D003C0A5DD400000000", "DISPLAY_NAME":"<idclip>", "NOTE":"<idclip>", "VISIBILITY_AREA_CONDITION":"{"data": {"link_operator":`

Attribute	Data type	Description	Example
		} }	"and","children":[]}}", "IS_SYSTEM":1 } } }
	-	**Format №2.** Relevant for user management events.	

		{
		“old”:{
		“<FIELD>”: “<VALUE>”,
		“<FIELD>”: “<VALUE>”
		}
		“new”:{
		“<FIELD>”: “<VALUE>”,
		“<FIELD>”: “<VALUE>”
		}
		}
			{
			"old":{
			"EMAIL":"asdasd@asdasd.ru",
			"CHANGE_DATE":"01-07-2014 09:46:29.000000"
			},
			"new":{
			"EMAIL":"asdasd11@asdasd.ru",
			"CHANGE_DATE":"01-07-2014 09:46:44.000000"
			}
			}

	-	**Format №3.** Relevant for log in and log out events. -	

		{
		"request":{
		"hostname": “<HOST NAME>”,
		"ip":"<IP-ADDRESS>",
		"login":"<USER LOGIN>"
			{
			"request":{
			"hostname":*null*,
			"ip":"127.0.0.1",
			"login":"officer"
			}

Attribute	Data type	Description	Example
		`}` `}`	`}`

9.11.2 Configuring TM Side

To configure the transfer of consolidated information from Traffic Monitor:

1. Create a database account for SIEM (see *Creating Siem User*).
2. Configure data transfer from Traffic Monitor to SIEM (see *Forwarding Logs to SIEM*).
3. If necessary, modify logging settings for Traffic Monitor database user session audit (see *Log Management for TM DB User Sessions*).

> **Note:**
>
> To change SIEM settings, use *SIEM integration* manager, which you can run in the command line with `iwtm-setuptool.py` script. To navigate in the settings manager, use keyboard arrows and **Enter** key or a mouse. To return to a previous window of the manager, press left key. To quit the manager, press **Ctrl+C**. For details on working in the settings manager, see the articles specified above.

Managing Siem User

This chapter describes creating, deleting and changing password for **siem** user, which is used by SIEM to interact with Traffic Monitor:

- Creating Siem User;
- Changing Siem User Password;
- Deleting Siem User.

Creating Siem User

To create siem user:

1. Run the command to start SIEM integration manager:
   `/opt/iw/tm5/bin/config/iwtm-setuptool.py`
2. Select **DB siem user** option.
3. Select **Create DB siem user** option.
4. Set parameters:

Parameter	Description
`DB admin login`	Login name for Oracle/PostgreSQL administrator account **Note:** the following login names are used: • `sys` for Oracle • `postgres` for PostgreSQL

Parameter	Description
`DB admin password`	Password for Oracle/PostgreSQL administrator account
`DB siem user password`	Password for new siem user

5. Press **Create**.

Changing Siem User Password

To change siem user password:

1. Run the command to start SIEM integration manager: `/opt/iw/tm5/bin/config/iwtm-setuptool.py`
2. Select **DB siem user** option.
3. Select **Change password for DB siem user** option.
4. Set parameters:

Parameter	Description
`DB admin login`	Login name for Oracle/PostgreSQL administrator account **Note:** the following login names are used: • `sys` for Oracle • `postgres` for PostgreSQL
`DB admin account`	Password for Oracle/PostgreSQL administrator account
`New siem user password`	New password for siem user

5. Press **Change password**.

Deleting Siem User

To delete siem user:

1. Run the command to start SIEM integration manager: `/opt/iw/tm5/bin/config/iwtm-setuptool.py`
2. Select **DB siem user** option.
3. Select **Delete DB siem user** option.
4. Set parameters:

Parameter	Description
`DB admin login`	Login name for Oracle/PostgreSQL administrator account
`DB admin password`	Password for Oracle/PostgreSQL administrator account

5. Press **Delete**.

Forwarding Logs to SIEM

To configure forwarding entries from TM Linux-server log files to SIEM:

1. Run the command to start SIEM integration manager:
   `/opt/iw/tm5/bin/config/iwtm-setuptool.py`
2. Select **Log messages forwarding configuration** option.
3. Set parameters for logs transfer:

Parameter	Description
`Enable forwarding`	Enable/disable forwarding logs to SIEM. Possible values: `Yes/No`
`Forwarding server`	SIEM server IP address or DNS name
`Forwarding server port`	Port of SIEM server where **syslog** is running
`Forwarding protocol`	Protocol of forwarding data to SIEM. Possible values: `TCP /UDP`
`Log messages severity`	Minimal level of log messages sent to SIEM server. Possible values: `Emergency`, `Alert`, `Critical`, `Error`, `Warning`, `Notice`, `Info`, `Debug`
`Size (MB)`	Acceptable size of the queue of messages to be sent to SIEM. Default value: `500`

4. Press **Ok**.

> **Warning!**
>
> The System does not log disconnection of users with database administrative permissions and therefore does not forward this information to SIEM.

Log Management for TM DB User Sessions

You can enable or disable logging of TM database user sessions in the OS log via **rsyslog** utility.

To enable logging of TM database user sessions:

1. Run the command to start SIEM integration manager:
   `/opt/iw/tm5/bin/config/iwtm-setuptool.py`
2. Select **DB audit logging management** option.
3. Enter database administrator credentials:
   - **DB admin login**: login name for Oracle/PostgreSQL administrator account;
   - **DB admin password**: password for Oracle/PostgreSQL administrator account.
4. Click **Check logging status**.
5. If:

- **Logging status: disabled**, click **Change logging status** to enable logging
- **Logging status: enabled**, click **Change logging status** to disable logging

> **Note:**
>
> For Oracle database, IWTM services are restarted when you change the logging status. Restarting services may take some time.

9.11.3 Type of Logs, Transmitted to SIEM

Rsyslog forms common system journal. It is a protocol, which is used to transmit logs to SIEM.

There are five major log types, transmitted to SIEM:

1. **iw_*** processes logs. For example:
   `Mar 11 14:32:30 iw-VMware-4224da3ab iw_expressd: 5 (18485:0x00007f53001b97e0) [INFO ] : <Root> Runtime environment initialized. Starting...`
2. **pguard** script logs. Script monitors state of services and restarts them when necessary:
   `Mar 11 14:32:26 iw-VMware-4224da3ab pguard: 148 (18256:0) [INFO] : expressd Terminating pid 18257 (signal 15)`
3. Logs of login to **Oracle DB** and logout from it:
   `Mar 11 14:28:20 iw-VMware-4224da3ab Oracle Audit[17786]: LENGTH : '153' ACTION :[7] 'STARTUP' DATABASE USER:[1] '/' PRIVILEGE :[4] 'NONE' CLIENT USER:[4] 'root' CLIENT TERMINAL:[13] 'Not Available' STATUS:[1] '0' DBID:[0]`
4. **Linux users** login and logout:
   `Mar 11 14:32:02 iw-VMware-4224da3ab runuser: pam_unix(runuser:session): session opened for user iwtm by root(uid=0)`
5. **Nagios** logs:
   `Feb 10 20:05:39 iw-VMware-4224289a4 nagios: SERVICE ALERT: IWTM;TM_DAEMONS_STATE;CRITICAL;SOFT;1;CRITICAL - updater(3)`

9.12 Deleting Temporary Files

Ability to delete temporary files has been implemented in the product as **clean_temporary_files.sh** script.

Script deletes the following information:

1. Files from operation system's temporary directory (`/opt/iw/tm5/tmp/` directory).
2. Data from Traffic Monitor file queue directory (`/opt/iw/tm5/queue/` directory).

> **Note:**
>
> New events will not be processed by the system during temporary files deletion. Information about all unprocessed data will be deleted at the moment of script execution.

To delete temporary files:

1. Run **clean_temporary_files.sh** script:
   `/opt/iw/tm5/bin/clean_temporary_files.sh`
2. Accept temporary files deletion:
   `yes`

10 Configuring Single Sign-On in InfoWatch products

Single Sign-On feature enables you to switch between InfoWatch Traffic Monitor and Investigation Centre without re-entering login and password. Single Sign-On works only when the transition between products is explicit and only for users created on the basis of the same Active Directory account.

To implement Single Sign-On:

1. On the main Investigation Centre server (`central` Platform node), type the following command in the command line:

   ```
   kubectl get secret guardkeys-central -n infowatch --output='go-template={{index .data
   "ec256-public.pem"}}' | base64 -d
   ```

   This will display a generated public key in PEM format. Key example:

   ```
   -----BEGIN PUBLIC KEY-----
   MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEIlNdAukpfJjZspZHWV2uu6U9D9/H
   IDCRYBmIRApbE4J2aiS3DGQICAJ3QShiFZtRYVATtJSPsyWGX4CP46Dbxg==
   -----END PUBLIC KEY-----
   ```

2. Copy and save the key.
3. On Traffic Monitor server, connect to the database:
   - For PostrgreSQL DBMS:

     ```
     psql postgres iwtm
     ```

   - For Oracle DBMS:

     ```
     sqlplus iwtm@iwtm
     ```

4. Specify a list of trusted systems which can authorize in Traffic Monitor by their user accounts. Give a name to each trusted system and paste the corresponding public key. For example, run the command:

   ```
   insert into trusted_system(system_name, public_key)
   values('my system','-----BEGIN PUBLIC KEY-----
   MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEIlNdAukpfJjZspZHWV2uu6U9D9/H
   IDCRYBmIRApbE4J2aiS3DGQICAJ3QShiFZtRYVATtJSPsyWGX4CP46Dbxg==
   -----END PUBLIC KEY-----');
   commit;
   ```

5. Exit the database:

   ```
   exit
   ```

6. In Traffic Monitor Management Console, configure and run synchronization with Active Directory to import a user who is going to work with both Systems. Wait for synchronization to fully complete.
7. In **Users and roles** section, add a new user from LDAP, assign a role to view events and visibility scopes.

8. Log out of this user account.
9. On a separate browser tab, open Investigation Centre web interface.
10. Configure and run synchronization with Active Directory to import the user mentioned earlier.
11. On **Users** tab, add a new user and assign a role to them.
12. Log out and then log in to the new user's account.
13. Go to **Events** section and click to select one or several events from the list and press **View events**.

A new tab with **Events** section of Traffic Monitor Management Console will open and the System will automatically authorize you. Besides, audit log will hold an event of the user logging in from an external system.

11 Interaction with external systems

Traffic Monitor and external systems exchange data via API. To make working with API more convenient, `iwtm-apidocs-publicapi` is implemented. This package is a part of Traffic Monitor distribution kit and needs to be installed. To install the package, run the following command on the Traffic Monitor server:

```
yum install iwtm-apidocs-publicapi
```

This command installs the following packages:

- `iwtm-apidocs-publicapi` that contains OpenAPI schemes for the public API;
- `iwtm-apidocs-common` that contains UI with a full description of methods including models, response codes, request parameters. To access this UI, use the address `<server_name_TM>/doc/`.

Further you can find descriptions of data exchange protocols that are used between Traffic Monitor and external systems:

- Uploading events (PushAPI SDK);
- Uploading Events via HTTP Protocol;
- Uploading sample documents and DB unloadings (REST API SDK);
- Access to InfoWatch Traffic Monitor events for external systems (DataExport API SDK).

Traffic Monitor supports interactions with other external systems via InfoWatch adapters. For more information, see *InfoWatch Traffic Monitor. Integrations Guides*.

11.1 Uploading events (PushAPI SDK)

This programming interface is designed to integrate third-party components with Traffic Monitor, which intercept data from various types of channels to process data analysis.
Third-party components form the data interception events and pass them to Traffic Monitor. Traffic Monitor classifies the received events, analyses them and assigns a verdict based on configured data protection policies. The events and verdicts can be viewed in Traffic Monitor console.

11.1.1 Principles of using PushAPI SDK programming interface

To configure sending events to Traffic Monitor, the following principles should be taken into consideration:

- Interception events are passed to Traffic Monitor in the form of *Event* object;
- Each *Event* has a set of required attributes specified by the user of the programming interface;
- Each *Event* may have a set of optional attributes specified by the user of the programming interface;
- Each *Event* has a *Sender* and a *Recipient*;
- Each *Sender* or *Recipient* may have a list of associated *Contacts* which are used to uniquely identify this sender/recipient;
- Each *Sender* or *Recipient* may have a list of attributes associated with this sender/recipient;each *Event* has associated *Data*, that is a set of bytes and/or text characters which represent the contents of the intercepted event;
- Each element of the *Data* should have a set of data associated with this element.

All possible *Events* that can be received by Traffic Monitor and analyzed are divided into groups (event classes) united by common characteristics. The event class defines the business logic of the event processing and how the event will be displayed in Traffic Monitor console.

You cannot create your own data types, that is event classes, in Traffic Monitor. When using SDK, select classes you need from the list of supported classes:

Event class / Service name in Traffic Monitor	Description
Email / E-mail	Any events of sending emails including emails with attachments. The event is characterized by a list of senders and recipients (persons), a sending date, a subject, a message body, and attachments. Includes corporate email (IMAP, POP3, SMTP, MAPI, NRPC) and web email (HTTP, HTTPS)
Chat / Messenger	Any events of exchanging only text messages or files within a single application. The application can be either a thin client on workstations and mobile devices or a web application. The event is characterized by a list of remarks. For each remark, the sender (person), the time of sending, and the message text must be specified. The following services are preinstalled in this class: Mobile communication, WhatsApp, Yahoo!, MS Linc, ICQ, Mail.Ru Agent, XMPP, Skype (SMS and text messages transferring)
Printers and MFPs / Printer and MFD	Events of data conversion on specialized devices for input/output of images and documents (e.g., printers, scanners). The event is characterized by a sender (person), a recipient (device), a number of copies, and a file with the original document/document image/print task. The class has a preinstalled service – *Printer*
Internet activity / Web activity	Events of sending requests from web browsers and similar programs via HTTP/ HTTPS protocols. Data for which no other class is selected also refers to this class. Sending a web message can be an example of an internet activity. The event is characterized by a sender (person) and a recipient (resource) for POST requests or a recipient (person) and a sender (resource) for GET requests. It is also characterized by the body of a request without precise data structure. The class has a preinstalled service – *Internet application*
Photo / Recording media	Events of photographing using device camera. The event is characterized by a sender (person) and a recipient (device). It is also characterized by an intercepted image. The class has a preinstalled service – *Photo camera*
File exchange / Messengers (if recipient and sender are persons) or *File exchange* (if at least one recipient or sender are not persons)	Any event of transferring files between any senders and recipients. The event is characterized by a sender (person), a recipient (person/device/ resource), and the intercepted file. Includes copying data to removable USB devices, cloud storage, sharing files via FTTP protocol.

Event class / Service name in Traffic Monitor	Description
	The following services are preinstalled in this class: FTP, External device, Mobile communication, WhatsApp, Yahoo!, MS Linc, ICQ, Mail.Ru Agent, XMPP, Skype
Call / Messenger	Any event of voice or video call via messenger or phone. The event is characterized by a person-sender (call initiator) and persons-recipients. It is also characterized by a file and a record time. Example: sending/receiving SMS or MMS or intercepting calls in social media. The class has the following preinstalled services: Mobile communication, MS Lync, Mail.Ru Agent, XMPP
Recording media / Recording media	Event of recording audio or video using a device. The event is characterized by a person-sender, a device-recipient, and an intercepted image. Example: microphone or dictaphone recordings, videocamera recordings and any events of photographing potentially confidential documents using device camera

The SDK user should define a class of object intercepted by SDK and a *Service* of intercepted object. *Service* determines what exactly was intercepted from the point of view of the user service, which the *Sender* used for transferring data. To add your own *Services*, use a third-party component registry file (see *Registering external component with PushAPI SDK* for details). The licensing policy in Traffic Monitor is implemented based on the object *Protocol* and the interceptor vendor identifier. You can also define an intercepted event *Service* as one of conditions when stating a Traffic Monitor policy or for analytical queries.

Based on event classes, you can define and register your service or select one from the Traffic Monitor built-in classes.

Examples

- There is a Traffic Monitor built-in service named *file_copy_removable*. Select it to intercept events of the *File exchange* class which occur when copying file to external devices;
- If SDK user has developed an interceptor for the Viber app, the user can define and register the service as *Viber* for interception of events like *Chat*, *Call*, *File exchange*. For each service of a single vendor, a separate license is issued.

For each *Event*, *Sender(s)* and *Recipient(s)* must be defined.
The following types of *Senders* and *Recipients* are supported:

- **Person** is a *User* who is defined as a creator, a sender or a recipient of the intercepted object. For the person identification, a set of person's contacts can be defined. The contact type can be selected from the list which is supported by SDK. An example of the contact type can be email, phone number or a Windows account;
- **Workstation** is a computer, a mobile device or a terminal server, where the intercepted object was created. For this type of a *Sender*, a list of *Contacts* can be defined;
- **Resource** is an address on the Internet to which data is passed;
- **Device** is a device that receives data when files are copied to removable devices.

The list of contacts associated with a *Person* makes it possible to identify the person by the contact. So, any intercepted action of a *Person* where at least one contact of the person is defined, allows Traffic Monitor to identify a particular user.

The SDK interface allows you to create logical connections between different objects, for example, a sender *Person* with a sender *Workstation*. This allows you to accurately determine the context of an intercepted operation, that is to determine who has initiated the data transferring and where.

The SDK interface enables sending data to a network in parts. It allows external developers to define, how to limit the load on the network.

A general algorithm of passing event the SDK interface:

1. Create an *Event* object and set its attributes.
2. Create senders/recipients of the event and specify their contacts and attributes.
3. Create the event data and set its attributes.
4. Establish connection with Traffic Monitor.
5. Check access to Traffic Monitor and a plugin license of an external developer.
6. Pass the event metadata.
7. Pass contents of the event data in parts.
8. Close the connection.
9. Delete the event object.

For more details on the process of passing events, see *Description of PushAPI SDK samples*.

11.1.2 General description of PushAPI SDK programming interface

The programming interface is implemented as a Thrift service. For more information on Thrift services, see *Apache Thrift*.

To use PushAPI SDK, install the latest version of Thrift.

For operating systems like Red Hat Enterprise Linux, you can use a package from a third-party repository named EPEL.

Example of installation Thrift on a computer running RHEL 7.x operating system:

1. Install the EPEL repository by running the command in the terminal:

```
yum install epel-release
```

2. Install Thrift by running the command:

```
yum -y install thrift
```

SDK samples include a Thrift scheme which describes a service implemented in Traffic Monitor. Below you can find a description of the scheme:

- Data Types in PushAPI SDK;
- Exceptions of PushAPI SDK;
- Traffic Monitor PushAPI SDK services;
- Contact types in Traffic Monitor PushAPI SDK;
- Traffic Monitor PushAPI SDK supported attributes;
- EventProcessor service in PushAPI SDK.

Data Types in PushAPI SDK

Code	Description
Identity structure	
`typedef i32 ItemId`	Item identifier in the event
`enum IdentityItemType { kPerson, kWorkstation, kDevice, kResource, }`	Identification item type
`struct Attribute { 1: required string name, 2: required string value }`	A separate attribute of any event entity (the event itself, identification, dataflow): • Contact; • Contact metadata
`struct ContactWithMeta { 1: required Attribute contact, 2: optional string meta }`	Identification contacts with metadata: • Contact; • Contact metadata
`struct Identity { 1: required ItemId identity_id, 2: required IdentityItemType identity_type, 3: required list<Attribute> identity_contacts, 4: optional list<Attribute> identity_attributes, 5: optional list<ContactWithMeta> identity_contacts_with_meta }`	Identification of the object receiver or sender: • A unique identification number inside the object, used for links to identification; • Identification item type; • Identification list of contacts; • An optional set of identification attributes; • The list of identification contacts with metadata
`struct EventData { 1: required ItemId data_id, 2: optional list<Attribute> data_attributes }`	Description of the event dataflow: • A unique identifier of dataflow inside the event, used for links to dataflow; • An optional set of the dataflow attributes
`struct ChatMessage { 1: required ItemId sender_id, 2: required string sent_time, 3: required string utf8_text,`	Description of a single message of the *Chat* class: • The link to the identification that is the message sender;

Code	Description
`4: optional ItemId mes_data_id }`	• Local time when the message was sent in `YYY-MM-DDTHH:MM:SS+HH:MM` format (ISO 8601 format yyyy-MM-ddTHH:mm:sszzz); • The message text in UTF-8 encoding; • An identifier of the object data that will be used to transfer the message content if `utf8_text` is empty

`typedef string LinkNameType`
`struct ItemLink {`
`1: required ItemId id,`
`2: required ItemId link_to_id,`
`3: required LinkNameType link_name`
`}`
	Description of connection between different entities of the event:
	• An identifier of the event entity, from which the connection goes;
	• An identifier of the event entity, to which the connection goes;
	• The name (type) of the connection

`const LinkNameType link_name_ws = "workstation_link"`	The list of names of supported connections between entities: • A name to establish connection between identifications of the *Person* and *Workstation* types

`struct Event {`
`1: required EventClass evt_class,`
`2: required TmServiceType evt_service,`
`3: optional list<Attribute> evt_attributes,`
`4: required list<Identity> evt_senders,`
`5: required list<Identity> evt_receivers,`
`6: required list<EventData> evt_data,`
`7: optional list<ChatMessage> evt_messages,`
`8: optional list<ItemLink> evt_links,`
`9: optional Identity evt_source,`
`10: optional Identity evt_destination`
`}`
	Description of the event that must be transferred to Traffic Monitor:
	• The event class;
	• The event service;
	• The event attributes;
	• The event senders;
	• The event recipients;
	• The event data;
	• Event messages for the *Chat* class;
	• Connections between the event entities;
	• The event source;
	• The event receiver

`struct Credentials {`
`1: required string company_name,`
`2: required string token`
`}`
	The structure that describes parameters to access Traffic Monitor:
	• The client name (or the company name in our case). The parameter connects an external developper's plugin with the license;

Code	Description
	• An authorization token. The token value is known only by the plugin and Traffic Monitor
`typedef i64 EventId`	An identifier of the event being loaded
`typedef i64 StreamId`	An identifier of the event dataflow being loaded
`typedef i32 InterfaceVersion` `const InterfaceVersion pushapi_version = 1`	PushAPI interface version

`enum EventClass {`
`kChat,`
`kEmail,`
`kMfp,`
`kWeb,`
`kFileExchange,`
`kPhoto,`
`kMultimedia,`
`kVoiceTalk`
`}`
	Supported event classes in PushAPI:
	• *Chat*;
	• *Mail*;
	• *Printing, copying and scanning*;
	• *Internet activity*;
	• *File exchange*;
	• *Photo*;
	• *Multimedia recording*;
	• *Call*

Note:

When sending events to Traffic Monitor by means of PushAPI, you can use any event class in order to be able to block data transfer based on analysis results. For this, Traffic Monitor allows you to set the **Block** verdict to events in the Data Protection Policy. See *Configuring Verdict Management* for details.

Exceptions of PushAPI SDK

Code	Description

`exception EventNotFound {`
`1: string message,`
`}`
	`EventId` specified in the list of parameters is not found in the current session of passing the event

`exception DataNotFound {`
`1: string message,`
`}`
	`DataId` for the specified `EventId` is incorrect, that is specified `DataId` is not found in the current session of passing the event

`exception StreamNotFound {`
`1: string message,`
`}`
	`StreamId` for the specified `EventId` is incorrect, that is specified `StreamId` is not found in the current session of passing the event

Code	Description
`exception InvalidEventFormat { 1: string message, }`	The event format is incorrect, that is validation for the contents of the event attributes failed. Each class has a set of required attributes
`exception InvalidCredentials { 1: string message }`	Incorrect parameters of access to the interface. Specify a correct token
`exception LicenseError { 1: string message }`	Traffic Monitor server does not have a correct license for passing events

Note:

Parameter `message` specifies the reason of exclusion.

Traffic Monitor PushAPI SDK services

Services embedded in Traffic Monitor which can be used for events:

```
const TmServiceType service_email = "email" //!< Email
const TmServiceType service_email_web = "email_web" //!< Web email
const TmServiceType service_im_icq = "im_icq" //!< ICQ
const TmServiceType service_im_skype = "im_skype" //!< Skype
const TmServiceType service_im_xmpp = "im_xmpp" //!< XMPP
const TmServiceType service_im_mmp = "im_mail_ru" //!< MMP
const TmServiceType service_im_lync = "im_lync" //!< MS Lync
const TmServiceType service_web = "web_common" //!< Web-messages
const TmServiceType service_file_removable_storage = "file_copy_removable" //!< File exchange with an
external device (removable device)
const TmServiceType service_file_ftp = "ftp" //!< File exchange through FTP
const TmServiceType service_print = "print" //!< Printing
const TmServiceType service_cloud_storage = "cloud_storage" //!< File exchange with a cloud storage
const TmServiceType service_terminal_session = "terminal_session" //!< File exchange through terminal
session
const TmServiceType service_network_resource = "network_resource" //!< File exchange with network resources
```

Service name	Classes	Description
`email`	Email	A corporate email interception service. The service does not depend on the used protocol or the mail server type. If you extend the *Email* class, we recommend you adding the

Service name	Classes	Description
		`email_` prefix to the service name, e.g., `email_google`
`email_web`	Email	A service for intercepting Web email
`im_icq`	Chat, Voice chat, File exchange	A service for intercepting ICQ messaging program (OSCAR protocol)
`im_skype`	Chat, Voice chat, File exchange	A service for intercepting Skype messaging program
`im_xmpp`	Chat, Voice chat, File exchange	A service for intercepting any messaging program that works on XMPP protocol (Jabber)
`im_mail_ru`	Chat, Voice chat, File exchange	A service for intercepting any messaging program that works on MMP protocol (Mail.ru Agent)
`im_lync`	Chat, Voice chat, File exchange	A service for intercepting MS Lync messaging program
`web_common`	Internet activity	A service for intercepting unparsed POST request
`file_copy_removable`	File exchange	A service for creating copies of files which are copied to removable storage devices
`ftp`	File exchange	A service for creating copies of files which are copied to removable storage devices
`print`	Printers and MFPs	A service for creating images of documents printed on a printer
`cloud_storage`	File exchange	A service for creating copies of files which are copied to cloud storages
`terminal_session`	File exchange	A service for creating copies of files which are copied via terminal session
`network_resource`	File exchange	A service for creating copies of files which are copied to network resources
Note: When you add a new service which is a messaging program, we recommend you adding `im_` prefix to the service name. For example, `im_viber`		

Note:

Services having other types of events can be registered using plugins.

Contact types in Traffic Monitor PushAPI SDK

Contact types, which Traffic Monitor business logic supports:

Code	Description
`const TmContactType contact_type_email = "email"`	Email address. Is used for identifications with *Person* type
`const TmContactType contact_type_auth = "auth"`	Login in OS in login@domain format. Is used for identifications with *Person* type
`const TmContactType contact_type_dnshostname = "dnshostname"`	DNS name. Is used for senders with *Workstation* type
`const TmContactType contact_type_icq = "icq"`	ICQ number. Is used for identifications with *Person* type
`const TmContactType contact_type_skype = "skype"`	Skype login. Is used for identifications with *Person* type
`const TmContactType contact_type_phone = "phone"`	Phone number. Is used for identifications with *Person* type
`const TmContactType contact_type_ip = "ip"`	IPv4 address. Is used for senders with *Workstation* type
`const TmContactType contact_type_sid = "sid"`	Security Identifier in Windows OS, in a text format. Is used for identifications with *Person* type
`const TmContactType contact_type_webaccount = "webaccount"`	Account name on a network resource
`const TmContactType contact_type_lotus = "lotus"`	Mailing address in Lotus. Is used for identifications with *Person* type
`const TmContactType contact_type_domain = "domain"`	Domain

Traffic Monitor PushAPI SDK supported attributes

List of attribute names, supported in Traffic Monitor:

Code	Attribute description
`typedef string EventAttributeType`	**Event attributes:**
`const EventAttributeType event_attr_capture_date = "capture_ts"`	Event interception local time in `yyyy-MM-ddTHH:mm:sszzz` format (ISO 8601) (*)
`const EventAttributeType event_attr_capture_server_ip = "capture_server_ip"`	IP-address of interception server (able to differ different copies of the same plugin) (*)
`const EventAttributeType event_attr_capture_server_fqdn = "capture_server_fqdn"`	DNS-name of interception server (able to differ different copies of the same plugin) (*)
`const EventAttributeType event_attr_inteceptor_id = "interceptor_id"`	Object identificator within the interceptor
`const EventAttributeType event_attr_inteceptor_protocol = "protocol_mnemo"`	Protocol set for object transfer
`const EventAttributeType event_attr_print_copies = "print_copies"`	Number of copies (integer within a string)
`const EventAttributeType event_attr_destination_type = "destination_type"`	Receiver type
`const EventAttributeType event_attr_chat_id = "chat_id"`	Group chat identifier in events of the `Messenger` service
`const EventAttributeType event_attr_chat_title = "chat_title"`	Group chat name in events of the `Messenger` service
`typedef string DataAttributesType`	**Common attributes of data stream:**
`const DataAttributesType data_attr_mimetype = "mimetype"`	MIME-type of content (to be defined only for the nodes, that have been already reviewed by client)
`const DataAttributesType data_attr_size = "data_size"`	Amount of data, in case it is known prior
`const DataAttributesType data_attr_sent_date = "sent_date"`	Object sending date in `YYYY-MM-DDTHH:MM:SS+HH:MM` format.

Code	Attribute description
	Able to set up independently the time of event formation and the time of real data transfer
`const DataAttributesType data_attr_comment = "data_comment"`	Comment
	***Email* class attributes of data stream:**
`const DataAttributesType data_attr_email_subject = "subject"`	Email subject
`const DataAttributesType data_attr_email_is_encrypted = "encrypted"`	Sign of encryption (*true* or *false* lines)
`const DataAttributesType data_attr_email_is_signed = "is_signed"`	Digital signature sign (*true* or *false* lines)
	***Files exchange* class attributes of data stream:**
`const DataAttributesType data_attr_file_filename = "filename"`	Short name of file (*)
`const DataAttributesType data_attr_file_source_file_path = "source_path"`	Full path to event source file
`const DataAttributesType data_attr_file_destination_file_path = "destination_path"`	Full path to transferred or scanned file or full address of web resource
	***Mobile networks* class attributes of data stream:**
`const DataAttributesType data_attr_voice_duration_sec = "call_duration"`	Duration in seconds (integer within a string)
	***Pinter and MFP* class attributes of data stream:**
`const DataAttributesType data_attr_print_job_name = "print_job_name"`	Name of printing task

Code	Attribute description
`const DataAttributesType data_attr_print_copies = "event_attr_print_copies"`	Number of copies (integer within a string)
`typedef string IdentityAttrType`	**Common identification attributes:**
`const IdentityAttrType identity_attr_comment = "identity_comment"`	Comment
`typedef string TWorkstation`	***Computers* type identification attributes:**
`const IdentityAttrType identity_attr_ws_type = "workstation_type"`	Type of workstation:
`const TWorkstation ws_type_computer = "ws_type_computer"`	Workstation
`const TWorkstation ws_type_mobile = "ws_type_mobile"`	Mobile device
`const TWorkstation ws_type_terminal = "ws_type_terminal"`	Terminal server
	***Device* type identification attributes:**
`const IdentityAttrType identity_attr_device_name = "device_name"`	Device name (*)
`const IdentityAttrType identity_attr_device_type = "device_type"`	Device type
	***Device*, *Resource*, *Workstation* type identification attributes for resource and receiver:**
`const IdentityAttrType identity_attr_endpoint_name = "name"`	Device or resource name (*)
`const IdentityAttrType identity_attr_endpoint_path = "path"`	Path on specified resource or device (*)
`const IdentityAttrType identity_attr_endpoint_type = "type"`	Resource or device type. To be filled with mnemonics from the guide

Code	Attribute description
`const IdentityAttrType identity_attr_endpoint_device_id`	Removable device ID
`const string identity_attr_print_location = "print_location"`	Printer location
`const string identity_attr_print_port_name = "print_port_name"`	Port name of printer
`const string identity_attr_print_server = "print_server"`	Server name of printer
	Attributes of *Resource* type identifications:
`const IdentityAttrType identity_attr_resource_url = "res_destination_url"`	Resource URL (*)
`const IdentityAttrType identity_attr_resource_address = "res_destination_host"`	DNS or resource IP-address (*)
Note: (*) - required attribute	

Note:

If an external system sends and event to Traffic Monitor, and its attribute `destination_path` has several values, then the event will be saved in Traffic Monitor with the first of the values, all others casted away.

EventProcessor service in PushAPI SDK

Traffic Monitor pushAPI SDK includes the EventProcessor service. The service provides an API for creation and transfer of intercepted events and objects.

Code	Description
`InterfaceVersion GetVersion()`	Function of getting the current interface version
`void VerifyCredentials(1: Credentials cred) throws(1: InvalidCredentials ex);`	Auditing function of PushAPI SDK interface access parameters. Allows you to verify the data transfer before its transfer

Code	Description
`ventId BeginEvent (1: Event event, 2: Credentials cred) throws(1: InvalidEventFormat ex1, 2: InvalidCredentials ex2, 3: LicenseError ex3, 4: EventNotFound ex4, 5: DataNotFound ex5);`	Function initiating the transfer process of event interception to Traffic Monitor by means of PushAPI SDK, where: • `event` – filled in structure, containing all metadata describing the interception event. Attributes: sender, recipient, list of data streams, related to event; • `cred` – structure which allows to authorize the event transfer process. The function returns the event identifier from the data transfer session. You can transfer several events during one TCP session
`StreamId BeginStream (1: EventId event_id, 2: ItemId event_data_id) throws(1: EventNotFound ex1, 2: DataNotFound ex2);`	Function of data stream transfer initiation, related to interception event, where: • `event_id` – event identificator, related to this data stream. To be returned by `BeginEvent` function; • `event_data_id` – data transfer identificator. This identifier is a part of event metadata, in `evt_data` field. The function returns the loading stream identifier.
`void SendStreamData(1: EventId object id, 2: StreamId stream_id, 3: binary chunk) throws(1: EventNotFound ex1, 2: StreamNotFound ex2, 3: InvalidEventFormat ex3);`	Function transferring the clipboard as a part of data stream related to interception event, where: • `event_id` – event identificator, related to this data stream. To be returned by `BeginEvent` function; • `stream_id` – data stream identificator, where clipboard is a part of transferred data. To be returned by `BeginStream` function; • `chunk` – clipboard. SDK user should specify the buffer size to provide enough speed of events loading with no overload of network infrastructure
`void EndStream(1: EventId event_id, 2: StreamId stream_id) throws(1: EventNotFound ex1, 2: StreamNotFound ex2, 3: InvalidEventFormat ex3);`	Function which records the end of a data stream related to an interception event, where: • `event_id` – event identificator, related to this data stream. To be returned by `BeginEvent` function; • `stream_id` – data stream identificator, where clipboard is a part of transferred data. To be returned by `BeginStream` function

Code	Description
`string GetEventDatabaseId(1: EventId object_id) throws(1: EventNotFound ex1);`	Function of getting unique interception event identification within Traffic Monitor System, where: • `object_id` – event identifier returned by `BeginEvent`. This value can be used to refer to loaded event. The function can be called after successful completion of `BeginEvent` function
`void EndEvent(1: EventId event_id, 2: bool abort) throws(1: EventNotFound ex1, 2: InvalidEventFormat ex2, 3: LicenseError ex3);`	Function which records the end of a data stream related to event interception, where: • `event_id` – identifier of an event for which the transfer is finishing; • `abort` – flag of the end of event processing within Traffic Monitor. If the flag is set, the event is not saved to the database

11.1.3 Registering external component with PushAPI SDK

To integrate an external data interception module with Traffic Monitor system, perform the steps:

1. Register with InfoWatch company the identifier string of vendor company producing a component. This will link the vendor company with a license that will be installed in Traffic Monitor system.
2. Get a developer license linked with the company identifier. This will enable loading interception events from this vendor to Traffic Monitor. The developer license allows you to load any event types (services).
3. Create a plugin for external component registry. The file format is described below (see *PushAPI SDK third-party component registry file format*). By this moment, the data source names for sample unloading should be already defined, which is to be generated by the external component. The license obtained as per step 2 is added to the registry file.

   > **Warning!**
   >
   > It is not recommended to change service names in the future. Each service corresponds to event type in Traffic Monitor. If you change the service name, then a new event type is created. This can cause duplication of event types in Traffic Monitor Management Console. For details on how to resolve duplication, see *Updating the third-party component registry file*.
   >
   > In the third-party component registry file, the service name is specified in the `"ADDS_SERVICES"->"SERVICE_TYPE"->"SERVICE_MNEMO"` field.

4. Load the external component registry plugin to Traffic Monitor system via Traffic Monitor console interface. For details, see the *Administration - Plugins* section in *InfoWatch Traffic Monitor. User Guide.*
5. Get the InfoWatch licence for a new integration and load it to the system using Traffic Monitor web interface, the *Administration - Licences* section.
6. In the properties of the loaded external component, get an access token and specify the token in settings of your external integration.

Now the System is ready to receive events from the specified vendor company with the specified interception service.

You can find the information on how to modify the registry file of the already registered third-party component in the *Updating the third-party component registry file* article.

PushAPI SDK third-party component registry file format

The plugin is an archive in a ZIP format. The archive includes:

- The **licenses** folder that contains license files
- The **icon** folder that contains files with pictograms for registered events
- The **manifest.json** file with information on a plugin

> **Note:**
>
> The **licenses** and **icon** folders are optional. License files and files with pictograms can be located in the root folder.

For external source systems, the **manifest.json** file should contain the following information:

Content	Description
`{`	
`"PLUGIN_ID": "",`	Unique plugin identifier (UUID) created by developer in accordance with RFC 4122 standard. It may include up to 40 characters (letters only in upper case, the "-" symbol cannot be used). The identifier is used for plugin update. Required. Type of data: string. **Example:** `"PLUGIN ID": "189C38D390396EB6E0530100007F1CA200000001",`
`"DISPLAY_NAME": "",`	Plugin name displayed. Required. Type of data: string. **Example:** `"DISPLAY_NAME": "Plugin name",`
`"DESCRIPTION": "",`	Plugin description displayed. Type of data: string. **Example:** `"DESCRIPTION": "Test plugin to demonstrate",`
`"VERSION": "",`	Plugin version. Required. Type of data: string. **Example:** `"VERSION": "1.0.0",`

Content	Description
`"IS_SYSTEM": ,`	"Preinstalled" attribute. Shows if the System includes the plugin. Type of data: boolean type. **Example:** `"IS_SYSTEM": false,`
`"VENDOR": "",`	Software company ID in accordance with the company name specified in the license. Required. Type of data: string. **Example:** `"VENDOR": "infowatch",`
`"LICENSE": [ { "PATH":"" } ],`	License files. The paths to license files should be specified in accordance with archive root folder. Required. Type of data: • `LICENSE` – array; • `PATH` – string. **Example** `"LICENSE": [ { "PATH":"licenses/license file 1.license" }, { "PATH":"licenses/license file 2.license" }, ],`
`"PATTERN_SEARCH_LICENSE": "",`	Template to search for previously uploaded licenses to be binded to plugin. It has the following format: `{operator:or|and|not,child[{name:value}]}"` Type of data: string. **Example operator:or** `"PATTERN_SEARCH_LICENSE": { "operator": "or", "conditions": [{ "origin": "dm" }, { "origin": "dmmobile" } ] },`

Content	Description

	Example operator:and
	
	`"PATTERN_SEARCH_LICENSE": { "operator": "or", "conditions": [ { "operator": "and", "conditions": [ {"key1": "value1"}, {"key2": "value2"}, {"key3": "value3"} ] }, { "operator": "and", "conditions": [ {"key1": "value1"}, {"key2": "value2"}, {"key3": "value3"} ] } ] }`
	
	Example operator:not
	
	`"PATTERN_SEARCH_LICENSE": { "operator": "and", "conditions": [ {"key1": "value1"}, {"key2": "value2"}, {"key3": "NONE"}, { "operator": "not", "conditions": [ {"key4": null} ] } ] }`

`"ADDS_EVENTS": { "OBJECT_TYPE": [`
 	
`{ "MNEMO": "", "SERVICE_MNEMO": "", "ICON": "", "LOCALE": { "": ""`
	Registered event types and protocols. Type of data: object.
	
	Array of registered event types containing service types they are related to. Type of data: array.
	
	Event type. Type of data: string.
	Service. Type of data: string.
	Files containing registered contact icons. Type of data: string.

Content	Description

`}`
`}`
`],`
	Localization of contact type names available at least in one language. Type of data: object.

`"PROTOCOL": [`
`{`
`"MNEMO": "",`
`"LOCALE": {`
`"": ""`
`}`
`}`
`]`
`},`
	Registered event types and protocols. Type of data: array.
	Required.
	Example
	`"ADDS_EVENTS": {`
	
	`"OBJECT_TYPE": [`
	`{`
	`"MNEMO": "cloud_storage",`
	`"SERVICE_MNEMO": "file",`
	`"LOCALE": {`
	`"rus": "Облачное хранилище",`
	`"eng": "Cloud storage"`
	`}`
	`},`
	`{`
	`"MNEMO": "im_telegram",`
	`"SERVICE_MNEMO": "im",`
	`"ICON": "icon/event_icon_telegram.png",`
	`"LOCALE": {`
	`"rus": "Telegram",`
	`"eng": "Telegram"`
	`}`
	`}`
	`],`
	`"PROTOCOL": [`
	`{`
	`"MNEMO": "telegram",`
	`"LOCALE": {`
	`"rus": "Telegram",`
	`"eng": "Telegram"`
	`}`
	`}`
	`]`
	`},`

`"USES_EVENTS": {` `"OBJECT_TYPE": [`	Types of existing events. Type of data: object. Array of registered event types specifying the types of services they are related to. Type of data: array.

`{`
`"MNEMO": "",`
`"PROTOCOL": [`
`{ "MNEMO": "" }`
`],`
	Type of event. Type of data: string.
	Array of protocols related to this type of event. Type of data: array.
	Protocols. Type of data: string.

Content	Description
`"ORIGIN": [ "" ] } ] },`	Array of interceptor type codes for InfoWatch plugins. Type of data: array. Codes. Type of data: string. Required. **Example** `"USES_EVENTS": { "OBJECT_TYPE": [{ "MNEMO": "email", "PROTOCOL": [{ "MNEMO": "pop3" }, { "MNEMO": "imap" }, { "MNEMO": "mapi" }, { "MNEMO": "smtp" }], "ORIGIN": ["dm", "dmmobile"] }, { "MNEMO": "email_web", "PROTOCOL": [{ "MNEMO": "http" }, { "MNEMO": "https" }], "ORIGIN": ["dm", "dmmobile"] }, { "MNEMO": "im_whatsapp", "PROTOCOL": [{ "MNEMO": "whatsapp" }], "ORIGIN": ["dm", "dmmobile"] }] },`

`"ADDS_SERVICES": { "SERVICE_TYPE": [`
 	
`{ "SERVICE_MNEMO": "", "DATA_CLASS": [""], "ICON": "", "LOCALE": { "": "", "": "" } }`
	Types of events registered via SDK services. Type of data: object.
	Types of intercepted services registered. Type of data: array.
	
	Name of intercepted service. Type of data: string.
	List of data categories intercepted by this service interceptor. Type of data: array of strings.
	Files containing registered contact icons. Type of data: string.
	Localization of contact type names available at least in one language. Type of data: object.
	
	Required.
	
	Example

Content	Description

]
},
	"ADDS_SERVICES": {
	"SERVICE_TYPE": [
	{
	
	"SERVICE_MNEMO": "newservice",
	"DATA_CLASS": ["kChat", "kFileExchange"],
	"ICON": "icon/TM_icon.png",
	"LOCALE":
	{
	"rus": "сервис#5",
	"eng": "service#5"
	}
	},
	{
	"SERVICE_MNEMO": "newservice2",
	"DATA_CLASS": ["kPhoto"],
	"ICON": "icon/TM_icon.png",
	"LOCALE":
	{
	"rus": "сервис#5+2",
	"eng": "service#5+2"
	}
	}
	]
	}

"OBJECT_HEADER": [	Custom event attributes. Type of data: array.
{ "NAME": "",	Unique attribute code to be used in the event headers and to be identified in xAPI/PushAPI. Type of data: string. If the developer's ID is not "IW", the code should start with the <Software company ID>_ prefix. **For example:** CISCO_CALL_DURATION
"NOTE": { "": "", "": "" },	Attribute name, displayed on Traffic Monitor management console. Type of data: object, string.
"TYPE": "", "FORMAT": "",	Type of value. Type of data: listing. Recording format. Possible values: number (integer, fractional), string, date and time UTC + specifying the time zone shifting, duration, hyperlink, listing, Boolean type. Type of data: listing.
"DATA_CLASS": [""], "USE_IN_POLICY": "", "USE_IN_NOTIFICATION": "", "USE_IN_QUERY": "", "USE_IN_LIST": "", "USE_IN_SHOW": "",	Services, whose events will be equipped with the new custom attribute. Type of data: strings array. Using attribute in policies. "1" - yes, "0" - no. Type of data: listing. Using attribute in tabular form of event list preview. "1" - yes, "0" - no.

Content	Description
`"USE_IN_DETAIL": "",`	Type of data: listing. Using attribute in short form of event list preview. `"1"` - yes, `"0"` - no. Type of data: listing. Using attribute in detailed form of event list preview. `"1"` - yes, `"0"` - no. Type of data: listing.
`"IS MULTIPLE_VALUE": "",`	Multiple value. `"1"` - yes, `"0"` - no. Type of data: listing.

`"VALIDATION_RULE": [ { "type": "" }] } ]`	Rule for checking query and policy values entered by a user. Defines possible values of a custom attribute. Type of data: object. You can set the following data types:
	• Boolean type
	• String
	• Fraction
	• Float
	• Duration
	• Listing
	• Hyperlink
	• Regular expression
	To set a regular expression, use the Perl Compatible Regular Expressions (PCRE) format.
	Example of the VALIDATION RULE field
	`"VALIDATION_RULE": [ { "type": "boolean", "value": ["yes","no"] } ]`
	`"VALIDATION_RULE": [ { "type": "string", "min": 1, "max": 100 } ]`
	`"VALIDATION_RULE": [ { "type": "integer", "min": 0, "max": 100 } ]`
	`"VALIDATION_RULE": [`

Content	Description

	<pre>{
	"type": "float",
	"min": 3,
	"max": 4
	}
	]
	
	"VALIDATION_RULE": [
	{
	"type": "duration",
	"value": [[1,31],[33,59]]
	}
	]
	
	"VALIDATION_RULE": [
	{
	"type": "enum",
	"enum": [
	"value1",
	"value2",
	"value3"
	]
	}
	]
	
	"VALIDATION_RULE": [
	{
	"type": "link",
	"link": "http://ya.ru"
	}
	]
	
	"VALIDATION_RULE": [
	{
	"type": "regexp",
	"pattern": "\d"
	}
	]</pre>
	Example of user attribute description
	<pre>"OBJECT_HEADER": [
	{
	"NAME": "header_multi_string",
	"NOTE": {
	"rus": "Заголовок строк1",
	"eng": "Header strings1"
	},
	"DATA_CLASS": ["kEmail"],
	"USE_IN_POLICY": "1",
	"USE_IN_QUERY": "1",
	"USE_IN_NOTIFICATION": "1",
	"USE_IN_LIST": "1",</pre>

Content	Description

	<pre>"USE_IN_SHOW": "1",
	"USE_IN_DETAIL": "1",
	"TYPE": "string",
	"FORMAT": "string",
	"IS_MULTIPLE_VALUE": "1"
	},
	{
	"NAME": "header_date",
	"NOTE": {
	"rus": "Заголовок дата1",
	"eng": "Header date1"
	},
	"DATA_CLASS": ["kEmail"],
	"USE_IN_POLICY": "1",
	"USE_IN_QUERY": "1",
	"USE_IN_NOTIFICATION": "1",
	"USE_IN_LIST": "0",
	"USE_IN_SHOW": "1",
	"USE_IN_DETAIL": "1",
	"TYPE": "string",
	"FORMAT": "date",
	"IS_MULTIPLE_VALUE": "0"
	}
	]</pre>

}	

Note

The `ADDS_EVENTS` and `USES_EVENTS` root fields are used only for InfoWatch plugins. For third-party plugins, use the `ADDS_SERVICES` field.

A full JSON scheme is a part of SDK samples (see *Description of PushAPI SDK samples*). A sample file of the module registration is included in the SDK.

Warning!

The file contains an expired license. To use the sample file for registration of a plugin, you need to obtain a new license.

Example of a Third-Party Registry File

Before you create a third-party component registry file, define the software development company identifier which corresponds to the company name. Register the identifier with InfoWatch and define the name of the service you are developing.

Below is the initial data for our example:

- Developer's ID: `MY_COMPANY`;

- Service name: `im_my_chat`, a service which intercepts data in an instant messaging application.

To create a third-party component registry:

1. Get the developer's license.
2. If necessary, select icons for the plugin and for the events it registers.
3. Describe the manifest file of the plugin.

To Get the Developer's License

Once you have defined the name of your service, contact InfoWatch to get a license for the plugin (see *InfoWatch Traffic Monitor. User Guide. Requesting License*). For the `MY_COMPANY` developer, the license file looks as follows:

```
"License": {
    "ActiveDays": 60,
    "Features": [
        {
            "common_name": "MY_COMPANY",
            "object_type": "im_my_chat",
            "protocol": "*"
        }
    ],
```

The `protocol` parameter can have the following values:

- `none` – no protocol
- The value of a certain protocol
- `*` – any protocol

To Describe the Manifest File of the Plugin

Fill in all the required root fields according to the initial data: `PLUGIN_ID`, `DISPLAY_NAME`, `VERSION`, `VENDOR`, `LICENSE` and `ADDS_SERVICES`.

The `im_my_chat` plugin intercepts data in an instant messaging application, so the value of the `DATA_CLASS` field in the `ADDS_SERVICES` root field is set to `kChat`. The plugin intercepts the *Chat* class events.

To describe the user's contact representation in intercepted events, add the `CONTACT_TYPE` field to the `ADDS_SERVICES` root field.

To add custom event attributes, use the `OBJECT_HEADER` construction. A custom attribute name should begin with the plugin developer name. The number of attributes is unlimited. Here, we are adding two attributes:

- `MY_COMPANY_FORWARDING_STATUS`, the message forwarding status. It shows if a message is original or forwarded.
  The attribute can have the following values:
  - `original`;
  - `forwarded`;
- `MY_COMPANY_ORIGINAL_SENDER`, the original sender. The attribute can have any value of the string type.

The manifest file of the plugin looks as follows:

```
{

   "PLUGIN_ID": "189C38D390396EB6E0540100007F1CA200000001",
   "DISPLAY_NAME": "Test plugin",
```

```
"DESCRIPTION": "Test plugin to demonstrate",
"VERSION": "1.0.0",
"IS_SYSTEM": false,
"VENDOR": "MY_COMPANY",
"ADDS_SERVICES": {
   "SERVICE_TYPE": [{
      "SERVICE_MNEMO": "im_my_chat",
           "DATA_CLASS": ["kChat"],
           "ICON": "icon/TM_icon.png",
           "LOCALE": {
                     "rus": "Новый сервис",
                     "eng": "New service"
           },
           "CONTACT_TYPE" : [{
                           "MNEMO" : "my_chat_Id",
                           "SCOPE" : ["person"],
                           "ICON": "icon/TM_icon.png",
                           "LOCALE": {
                                     "rus": "Контакт тестового плагина",
                                     "eng": "Test plugin contact"
                           }
                  }
           ]
   }]
},
"LICENSE": [
{ "PATH" : "licenses/tm_license_2023-07-24_.license" }
],
"OBJECT_HEADER": [{
     "NAME": "MY_COMPANY_FORWARDING_STATUS",
     "NOTE": {
             "rus": "Статус пересылки сообщения",
             "eng": "Forwarding Message Status"
     },
     "TYPE": "string",
     "FORMAT": "enum",
     "DATA_CLASS": ["kChat"],
     "USE_IN_POLICY": "1",
     "USE_IN_QUERY": "1",
     "USE_IN_NOTIFICATION": "1",
     "USE_IN_LIST": "1",
     "USE_IN_SHOW": "1",
     "USE_IN_DETAIL": "1",
     "IS_MULTIPLE_VALUE": "0",
     "VALIDATION_RULE": [{
                        "type": "enum",
                        "enum": [
                                "original",
                                "forwarded"
                        ]
}]
},
{
     "NAME": "MY_COMPANY_ORIGINAL_SENDER",
     "NOTE": {
             "rus": "Оригинальный отправитель",
             "eng": "Original sender"
     },
     "TYPE": "string",
```

```
        "FORMAT": "string",
        "DATA_CLASS": ["kChat"],
        "USE_IN_POLICY": "0",
        "USE_IN_QUERY": "0",
        "USE_IN_NOTIFICATION": "1",
        "USE_IN_LIST": "0",
        "USE_IN_SHOW": "1",
        "USE_IN_DETAIL": "1",
        "IS_MULTIPLE_VALUE": "0"
   }]
}
```

In our example, we locate the license file in the plugin registry file, which is in the **licenses** folder in a ZIP archive. Below you can see how to define it in the manifest file:

```
   "LICENSE": [
   { "PATH" : "licenses/tm_license_2023-07-24_.license" }
   ],
```

If the license file is uploaded to the Traffic Monitor separately in the **Manage** -> **Licenses** section, specify it in the plugin manifest file by defining the template that searches for the previously loaded licenses:

```
"PATTERN_SEARCH_LICENSE": {
    "operator": "and",
    "conditions": [{
            "common_name": "MY_COMPANY"
        }, {
            "object_type": "im_my_chat"
        }, {
            "protocol": "*"
        }
    ]
},
"LICENSE": []
}
```

`VENDOR` and `SERVICE_MNEMO` fields of the manifest file correspond to the `common_name` and `object_type` names in Traffic Monitor, so their values in the license search template are set to `MY_COMPANY` and `im_my_chat` correspondingly.

How to Work with the Plugin

Once you have received the license, selected icons and described the manifest file, create the third-party registry file and upload it to Traffic Monitor in the **Manage** -> **Plugins** section.

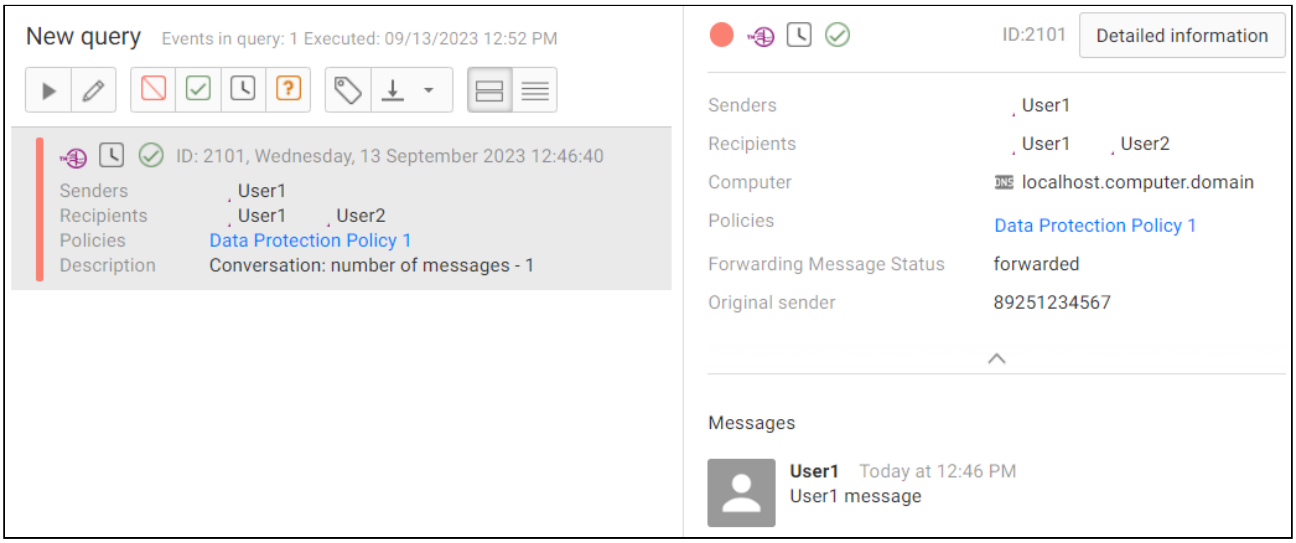
To start working with the registered plugin, use the `pushapi-util` utility. See *Description of PushAPI SDK samples* for more information about the utility.

To send a *Chat* class event of the `im_my_chat` service to Traffic Monitor, use the following command:

```
pushapi-util \
--host TM_host \
--token im_my_chat_token_value \
--id MY_COMPANY \
--class imchat \
```

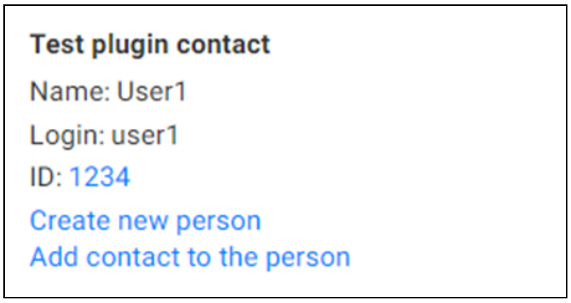
```
--service im_my_chat \
--evtattr event_name:"new service chat Event" \
--sender "my_chat_Id:1234:'{\"display_name\":\"User1\",\"login\":\"user1\",\"url\":\"https://my_chat.ru/
user1\"}'" \
--receiver "my_chat_Id:76543:'{\"display_name\":\"User2\",\"login\":\"user1\",\"url\":\"https://my_chat.ru/
user2\"}'" \
--mes "my_chat_Id:1234:'{\"display_name\":\"User1\",\"login\":\"user1\",\"url\":\"https://my_chat.ru/
user1\"}'","User1 message" \
--evtattr MY_COMPANY_FORWARDING_STATUS:forwarded \
--evtattr MY_COMPANY_ORIGINAL_SENDER:89251234567
```

In the Traffic Monitor Management Console, the event looks as follows:



You can see assigned custom attributes in the event description. They are not required, so if you do not define them in the code, you will not see them in the event.

We have defined the `CONTACT_TYPE` field in the manifest file, so the Management Console displays the user's contact in the event as follows:

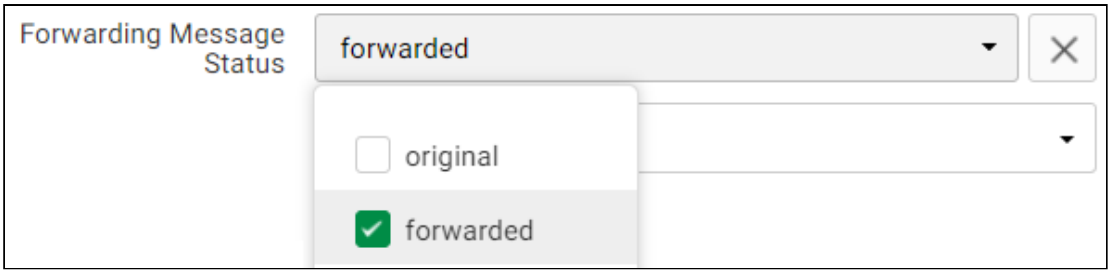


To Use a Custom Attribute in a Data Protection Policy

The `MY_COMPANY_FORWARDING_STATUS` custom attribute in the `USE_IN_POLICY` field is set to `1`. You can use this attribute to create a data protection policy.



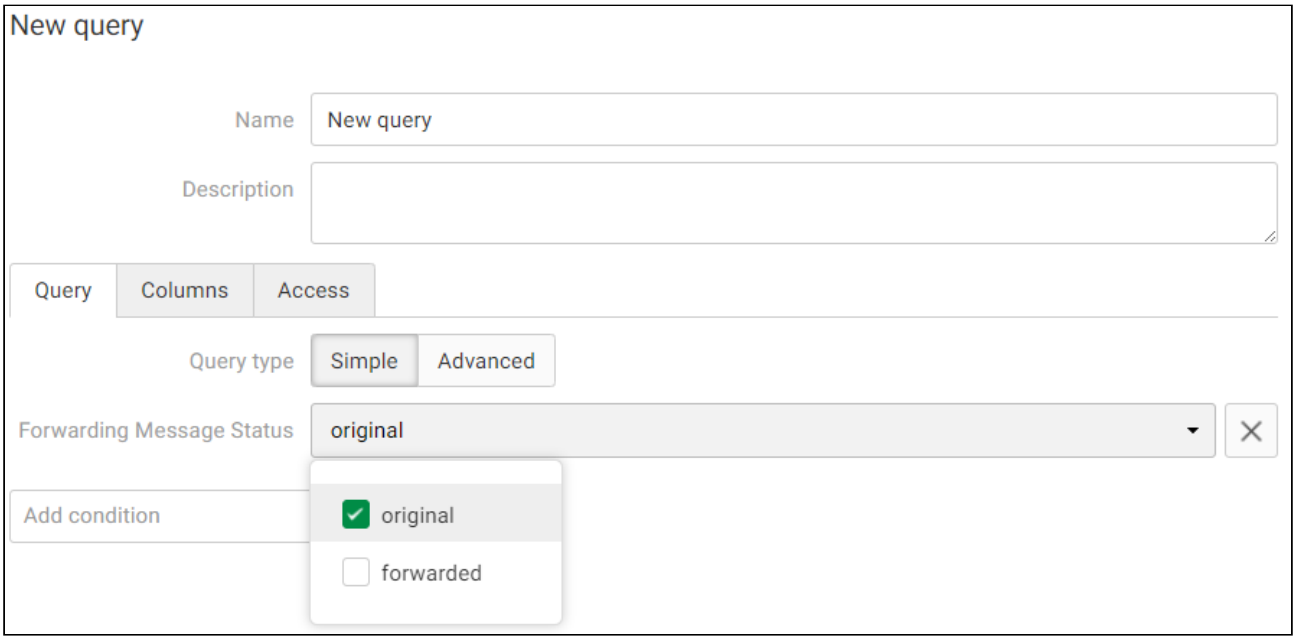
In the *Forwarding rules* parameters you can see the *Forwarding Message Status* attribute. To intercept forwarding messages, set the `forwarded` value for this attribute.



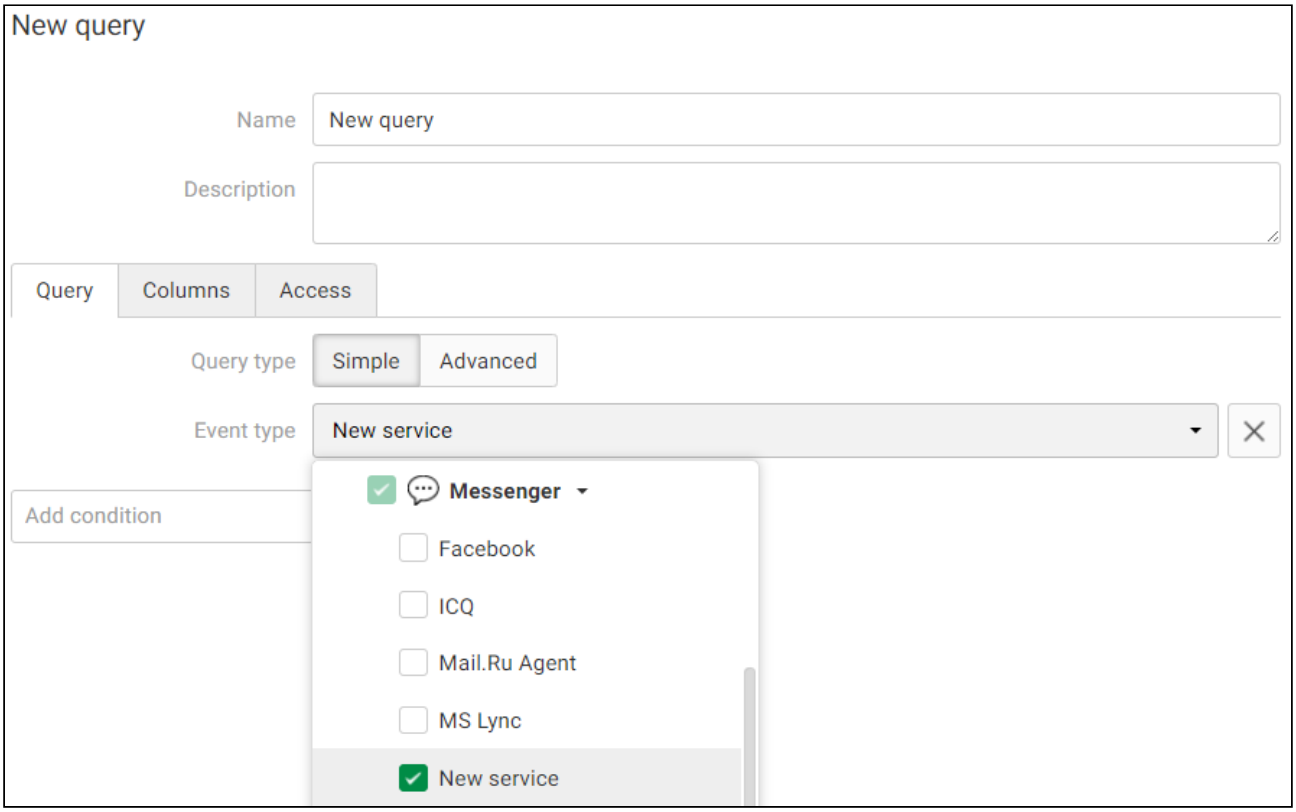
As a result, the policy responds to events with the `MY_COMPANY_FORWARDING_STATUS:forwarded` attribute.

To Create Queries with the New Service

The `MY_COMPANY_FORWARDING_STATUS` custom attribute in the `USE_IN_QUERY` field is set to `1`. You can use this attribute to create queries in the **Events** section.



You can also use the service itself to create queries. Select *New Service* as the event type.



Updating the third-party component registry file

To update the third-party registry file:

1. Make sure that, in Traffic Monitor, there is an already registered plugin with the same value of the `"PLUGIN_ID"` field as the new plugin has.
   You can find the information about all registered plugins in the **plugin** table of the database. For details on working with a database, see *Database Administration*.
2. Create the new version of the manifest file of the plugin. For this, make the necessary changes to the initial manifest file.

   > **Warning!**
   >
   > When updating plugin, it is not recommended to change service name, i.e. the value of the `"SERVICE_MNEMO"` field. Because, in this case, a new event type will be created in Traffic Monitor and the events intercepted by old plugin will not be associated with the plugin, that has a new value of the `"SERVICE_MNEMO"`.
   >
   > If you want to change the event type name that is displayed in the Traffic Monitor web interface, edit the `"LOCALE"` field.
   >
   > If you change the `"SERVICE_MNEMO"` field and, at the same time, not change the `"LOCALE"` field, then, in Traffic Monitor, there will be two event types with the same names. In Traffic Monitor Management Console, they cannot be distinguished from each other.

3. Upload the new plugin to the Traffic Monitor Management Console.

Deletion of duplicating event types

If you have added several event types with different service names (`"SERVICE_MNEMO"`) but with same displayed names (`"LOCALE"`) to Traffic Monitor, you can resolve duplication by deletion the information on irrelevant event types from the database. In doing so, you can delete not the entire plugin but only the event type created by it and caused the duplication.

Warning!

Perform deletion of information from the database carefully. Deletion information by mistake can cause incorrect work of the System.

Before deletion, it is recommended to create backup copies of database and tables.

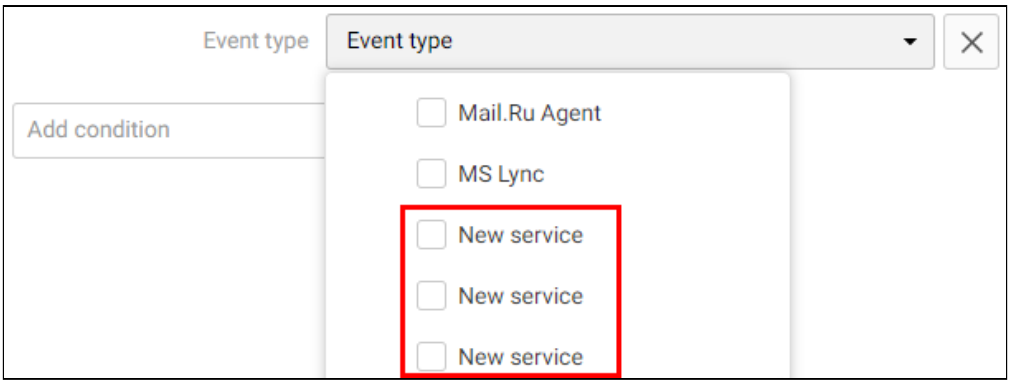
Warning!

Deletion of duplicating event type will cause incorrect displaying of events of this event type and security policies using this event type. Results of queries that must contain events of the deleted type will not be displayed.

If duplicating event type is used in Traffic Monitor, you can:

- Not delete it, but change the displayed name of the new event type by correctly updating plugin.
- Wait for storage period of the events of the duplicating type to expire. Then delete this event type from the database.

Let's look at the deletion procedure with the example of the plugin with the `"im_my_chat"` event type from the *Example of a Third-Party Registry File* article. To demonstrate, we have incorrectly updated the plugin twice. The `"SERVICE_MNEMO"` field had `"im_my_chat_new"` and `"im_my_chat_final"` values. The `"LOCALE"` field had the `"New service"` value in all cases. As a result, the duplication has occurred. For example, when you create a query in the **Events** section of Traffic Monitor Management Console, you can see three *New service* event types:



To delete duplicating event types:

1. In the **plugin** table of the database, find identifiers (ID) of the plugins that caused duplication. This identifiers are recorded in the **plugin_id** field.

In our example, there is only one record with such a plugin:

	plugin_id [PK] character varying (40)	display_name character varying (4000)	description character varying (4000)
1	[redacted]	InfoWatch Device Monitor	InfoWatch Device Monitor
2	4B8038D390396EB6E0540100007F1CA200000001	Test plugin final	Test plugin to demonstrate
3	[redacted]	InfoWatch Data Discovery	Предустановленный плагин...
4	[redacted]	InfoWatch Sample documen...	Предустановленный плагин...

2. Find identifiers of the event types and services created by these plugins. For this:
   a. In the **plugin_data** table, find all data for these plugins. You can do it using SQL query of the following type:

```
SELECT * FROM iwtm.plugin_data
WHERE plugin_id in ('4B8038D390396EB6E0540100007F1CA200000001')
ORDER BY plugin_id
```

   If there are several duplicating plugins, then specify their **plugin_id** field values in brackets, separated by commas.
   Result of the query in our example:

	plugin_id character varying (40)	entry_type character varying (255)	entry_value character varying (4000)
1	4B8038D390396EB6E0540100007F1CA200000001	CONTACT_TYPE_MNEMO	my_chat_Id
2	4B8038D390396EB6E0540100007F1CA200000001	CONTACT_TYPE_SCOPE	person
3	4B8038D390396EB6E0540100007F1CA200000001	OBJECT_TYPE_OBJECT_TYPE_ID	FAF6B5B261204B31BEED5F67798A477C00000000
4	4B8038D390396EB6E0540100007F1CA200000001	OBJECT_TYPE_SERVICE_ID	DD6ECB5227DA11E2B507CBDB6088709B00000000
5	4B8038D390396EB6E0540100007F1CA200000001	OBJECT_TYPE_MNEMO	im_my_chat_final
6	4B8038D390396EB6E0540100007F1CA200000001	object_header	my_company_forwarding_status
7	4B8038D390396EB6E0540100007F1CA200000001	object_header	my_company_original_sender
8	4B8038D390396EB6E0540100007F1CA200000001	license	8fee32283d89c06fafae1424d52871469ae0ee5235ade72a42d512ed2016b6...

   b. In the obtained data, for each **plugin_id** value, find those records where **entry_type** field value equals `OBJECT_TYPE_OBJECT_TYPE_ID` or `OBJECT_TYPE_SERVICE_ID`.
   Remember the pairs of the values of these fields. It is identifier of the event type created by duplicating plugin and identifier of the service.
   In the table from the example, one pair is found. It contains the following values:
   `OBJECT_TYPE_OBJECT_TYPE_ID = "FAF6B5B261204B31BEED5F67798A477C00000000"`
   `OBJECT_TYPE_SERVICE_ID = "DD6ECB5227DA11E2B507CBDB6088709B00000000"`
3. In the **object_type** table, find records with the pairs of identificators from the previous step. For this, you can use the following query:

```
SELECT * FROM iwtm.object_type
WHERE object_type_id = 'FAF6B5B261204B31BEED5F67798A477C00000000'
  AND service_id = 'DD6ECB5227DA11E2B507CBDB6088709B00000000'
```

	object_type_id [PK] character varying (40)	service_id character varying (40)	language [PK] character varying (3)	display_name character varying (4000)	mnemo character varying (256)
1	FAF6B5B261204B31BEED5F67798A477C00000000	DD6ECB5227DA11E2B507CBDB6088709B00000000	eng	New service	im_my_chat_final
2	FAF6B5B261204B31BEED5F67798A477C00000000	DD6ECB5227DA11E2B507CBDB6088709B00000000	rus	Новый сервис	im_my_chat_final

4. On the **mnemo** field, determine a pair of identifiers that refers to the relevant plugin.
   If there is the only one pair, then go to the 6 step.
   In our example, found records have `"im_my_chat_final"` value of the **mnemo** field. It means that these records refer to the relevant plugin.

5. For each pair that does not refer to the relevant plugin, perform the following actions:
    a. Make sure that, in the **object** table, there are no events of the event type from this pair. For this, run the query using the values of the pair of the identifiers – `OBJECT_TYPE_ID` and `SERVICE_ID`:

    ```
    SELECT count(*) FROM object
    WHERE object_type_code='<OBJECT_TYPE_ID>'
    AND service_code='<SERVICE_ID>'
    ```

    If the result of the query equals zero, then there are no events.

    > **Warning!**
    >
    > If, in the **object** table, there are events of this event type but you want to delete it anyway, it is recommended to delete all events of this type from this table first.

    b. From the **object_type** table, delete records related to this event type. These are the records found in the step 3.
    c. From the **plugin_data** table, delete records related to this event type. For instance, these can be:
        - event type identifier – `"OBJECT_TYPE_OBJECT_TYPE_ID"`
        - service identifier – `"OBJECT_TYPE_SERVICE_ID"`
        - event type name – `"OBJECT_TYPE_MNEMO"`
6. In the **object_type** table, find the records related to the duplicating event types. For this, run the following query:

    ```
    SELECT * FROM iwtm.object_type
    WHERE display_name = '<EVENT_TYPE_DISPLAY_NAME>'
    ```

    where `<EVENT_TYPE_DISPLAY_NAME>` – the displayed name of the event type.
    In our example, the query will be as follows:

    ```
    SELECT * FROM iwtm.object_type
    WHERE display_name = 'New service'
    ```

	object_type_id [PK] character varying (40)	service_id character varying (40)	language [PK] character varying (3)	display_name character varying (4000)	mnemo character varying (256)
1	76505E31A1DE46698339D6AC00DD0DE8000000...	DD6ECB5227DA11E2B507CBDB6088709B00000000	eng	New service	im_my_chat_new
2	083030E950F948B891CCE1B048A8E70300000000	DD6ECB5227DA11E2B507CBDB6088709B00000000	eng	New service	im_my_chat
3	FAF6B5B261204B31BEED5F67798A477C00000000	DD6ECB5227DA11E2B507CBDB6088709B00000000	eng	New service	im_my_chat_final

7. On the **mnemo** field, determine duplicating event types.
    In our case, `"im_my_chat_final"` is an event type of the relevant plugin. It means that `"im_my_chat"` and `"im_my_chat_new"` are old duplicating types which we want to delete.
8. For each duplicating event type:
    a. Make sure that, in the **object** table there are no event of this type. Follow a procedure similar to step 5a of this instruction.
    b. In the *object_type* find and delete all records related to this type. They differ from each other only in localization, i.e. in values of the **language** and **display_name** fields.
    For this, you can use the following query:

```
SELECT * FROM iwtm.object_type
WHERE mnemo = '<MNEMO>'
```

For instance, for the `"im_my_chat"` event type, the following records will be displayed:

	object_type_id [PK] character varying (40)	service_id character varying (40)	language [PK] character varying (3)	display_name character varying (4000)	mnemo character varying (256)
1	083030E950F948B891CCE1B048A8E70300000000	DD6ECB5227DA11E2B507CBDB6088709B00000000	rus	Новый сервис	im_my_chat
2	083030E950F948B891CCE1B048A8E70300000000	DD6ECB5227DA11E2B507CBDB6088709B00000000	eng	New service	im_my_chat

9. Run the procedure of updating database object list. For this:
   - If you use the PostgreSQL DBMS, run the command:

```
SELECT sp_populate_configuration('bookworm');
```

   - If you use the Oracle DBMS, run the command:

```
begin
    sp_populate_configuration('bookworm');
end;
```

10. Make sure that duplicating event types are absent in Traffic Monitor Management Console.

11.1.4 Description of PushAPI SDK samples

Samples are the source codes of console utilities, which implement loading events into Traffic Monitor using Thrift interface of PushAPI SDK.

Console utilities

- The **pushapi-cpp** utility is implemented in C++ and prepared for assembly in Windows and Linux operating systems. This utility demonstrates scripts of loading all event types that are already registered with Traffic Monitor.
- The **pushapi-csharp** utility is implemented in C# and prepared for assembly in Windows operating system. This utility demonstrates scripts of loading all event types that are already registered with Traffic Monitor.
- The **pushapi-demo.py** utility is implemented in Python. It is prepared for assembly in Windows and Linux operating systems. This utility demonstrates scripts of loading all event types that are already registered with Traffic Monitor.
- The **pushapi-util** utility is implemented in C++ and prepared for assembly in Windows and Linux operating systems. This utility has an advanced command-line interface that enables uploading any events to Traffic Monitor system, including events registered with an external system.

The pushapi-util utility command line description

```
./pushevent-util
--host <pushapi_address> [--port <pushapi_port>]
--token <plugin_token>
--id <company>
--class <event_class>
--service <event_service>
```

```
--evtattr <attr_name:attr_value,…, attr_name:attr_value>
--sender <contact_type:contact_value,attr_name:attr_value,…,attr_name:attr_value>
--receiver <contact_type:contact_value,attr_name:attr_value,…,attr_name:attr_value>
--data file:<data_falie_name>, attr_name:attr_value,…, attr_name:attr_value
--mes <contact_type:contact_value>,"message text"
[--addctx]
[--show_event]
```

Where:

- `<pushapi_address >` – PushAPI server address
- `<pushapi_port>` – PushAPI server port. If the port is not specified, a default value is used
- `<plugin_token>` – the plugin authorization token. Copy it from the section of the given plugin in the Traffic Monitor Console
- `<company>` – an identifier of the plugin vendor. A string, which must be the same as the string used for generating a license for the plugin
- `<event_class>` – the class of the passed event
- `<event_service>` – the service of the passed event. The plugin can register intercepted services by their names and pass the names;
- `--evtattr` – attributes of the passed event. Can be viewed in the list of commands with a sample. Some attribute names have fixed values and business sense
- `--sender` – the event sender. The first pair of parameters defines the sender's contact. Then the sender's attributes are specified.
  There are three special contact types:
  - dev – the sender/recipient is a device;
  - res – the sender/recipient is a resource;
  - ws – the sender/recipient is a "Workstation".

  If no sender is specified, then two senders will be generated: one sender with *Person* type and the *auth* contact type and the other sender with *Workstation* type and *dnshostname* contact. Only one sender can be specified

  Example:

  If the event sender has a VK account and meta-information is specified for the event:

  ```
  --sender "vkontakte:349051292:'{\"display_name\":\"some name\",\"url\":\"some_url\",
  \"login\":\"some_login\"}'"
  ```

- `--receiver` – the event recipient. Has the same parameters as the sender
- `--data` – the event data. The first pair of parameters defines the path to a file with the event body. Other parameters describe attributes of these data. For some event types, particular attributes are required
- `--mes` – passing events of *Chat* class. A contact of the message sender and the message contents are specified. There can be several messages
- `--addctx` – an optional parameter. Automatically adds two senders to the event: one sender with *Person* type and *auth* contact type and the other sender with *Workstation* type and *dnshostname* contact type;
- `--show_event` – an optional parameter. Outputs information about the Event structure to the console. The information is output in JSON format.

pushapi-util utility installation

To install the **pushapi-util** utility from the QATOOLS package:

1. On the workstation with Traffic Monitor installed, extract **pushapi-util** to the `/opt/iw/tm5/bin` catalogue. Find the `pushapi-util` utility in the `iwtm-qatools-x.x.x.xxx-release.xrpm/iwtm-qatools-x.x.x.xxx-release.x86_64.cpio/./opt/iw/tm5/bin` package, where `x.x.x.xxx` is the build number.

   > **Note:**
   >
   > If you extract **pushapi-util** to a different catalogue, add the catalogue to the environment variables:
   >
   > ```
   > export PATH=$PATH:<directory>
   > ```

2. Go to the catalogue with the extracted **pushapi-util** utility.
3. Make the utility file executable:

   ```
   chmod u+x pushapi-util
   ```

Find the examples on how to call this utility in the source codes of samples or in the *Examples on How to Use the pushapi-util Utility* article.

Building PushAPI SDK samples

Find the specifics of building PushAPI SDK samples in the following articles:

- Building samples under Linux in PushAPI SDK;
- Building samples under Windows in PushAPI SDK.

Besides, SDK samples include:

- **pluging.json** file – JSON scheme of `mainifest.json` file, which is included in the registration file for a new plugin
- **plugin.zip** file – a sample of file used for registration of an external plugin

> **Warning!**

The file contains an expired license. To use the sample file for registration of the plugin, you need to obtain a new license.

Examples on How to Use the pushapi-util Utility

With the help of pushapi-util test utility, you can send events for any built-in services and classes to Traffic Monitor using the built-in Device Monitor plugin.

> **Note**
>
> In commands, use Device Monitor token as **token** attribute value. Copy it from **Manage** -> **Plugins** section in Management console.

Examples on how to send events of *Email* class

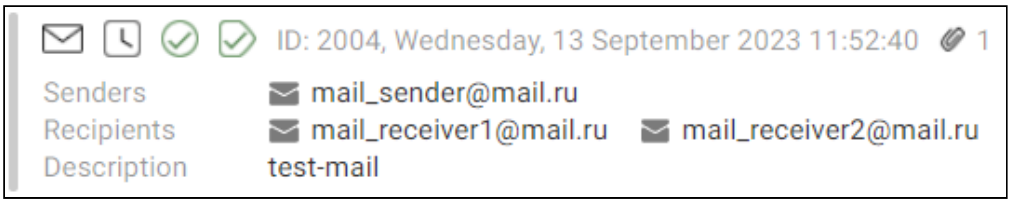
Example 1:

To send an *Email* class event of `Email` service to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class email \
--service email \
--evtattr event_name:"Send email" \
--sender email:"mail_sender@mail.ru" \
--receiver email:"mail_receiver1@mail.ru" \
--receiver email:"mail_receiver2@mail.ru" \
--data file:/root/test_email.eml
```

In the command above, `test_email.eml` is a file that contains the message. The `data`, `sender` and `receiver` attributes are required.

In Traffic Monitor Management console, the event looks as follows:



Example 2:

To send an *Email* event of `Web email` service to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class email \
```

```
--service email_web \
--evtattr event_name:"Send email_web" \
--sender email:"mail_sender@mail.ru" \
--receiver email:"mail_receiver1@mail.ru" \
--data file:/root/test_email.eml
```

Examples on how to send events of *Chat* class

Example 1:

To send a *Chat* class event for ICQ application to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class imchat \
--service im_icq \
--evtattr event_name:"ICQ chat event" \
--receiver icq:IcqReceiver \
--mes icq:IcqSender,"Confidential information"
```

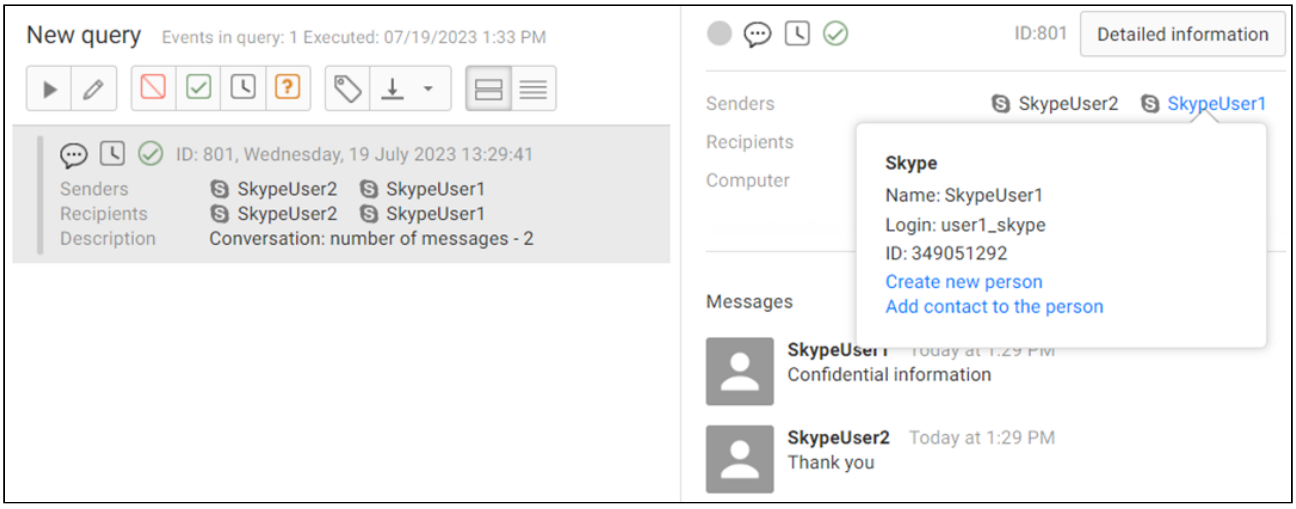
The `mes` and `receiver` attributes are required.

Example 2:

To send a *Chat* class event for Skype application to Traffic Monitor, use the command below. Note that in this example the event contains metadata about senders.

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class imchat \
--service im_skype \
--evtattr event_name:"Skype event" \
--mes "skype:349051292:'{\"display_name\":\"SkypeUser1\",\"login\":\"user1_skype\"}'","Confidential
information" \
--mes "skype:349051100:'{\"display_name\":\"SkypeUser2\",\"login\":\"user2_skype\"}'","Thank you"
```

The sender metadata includes the displayed name and Skype login name of a sender. In Traffic Monitor Management console, the event looks as follows:



Examples on how to send events of *Printers and MFPs* class

To send a *Printers and MFPs* class event of `Print` service to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class mfp \
--service print \
--evtattr event_name:"Print event" \
--receiver dev:"Some Printer Name" \
--data file:/root/test.png,filename:test.png
```

and

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class mfp \
--service print \
--evtattr event_name:"Print event" \
--receiver dev:"Some Printer Name",print_port_name:"Some print port name",print_location:"Some print
location" \
--data file:/root/test.png,print_copies:3,print_job_name:"Some print job name"
```

The `receiver` and `data` attributes are required. The command should also contain one of the following attributes: `filename` or `print_job_name`.

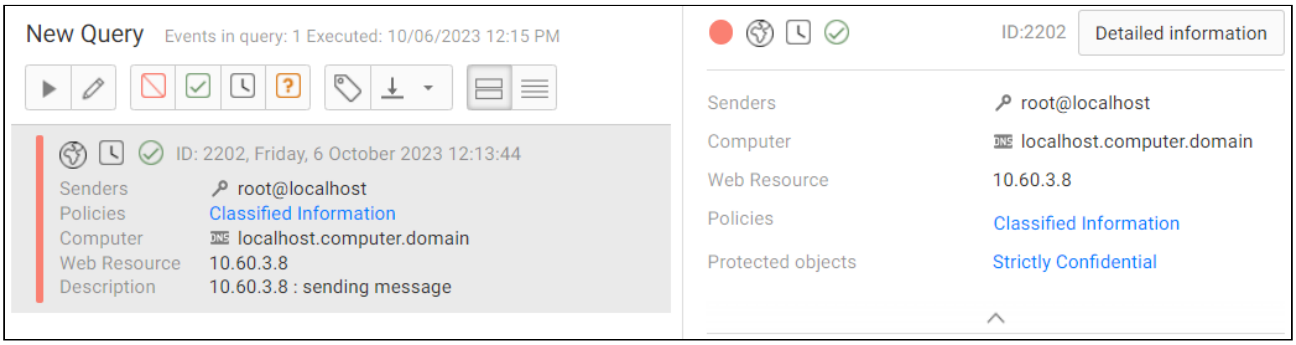
Example on how to send events of *Internet Activity* class

To send an *Internet Activity* class event of `Web_common` service to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
```

```
--id iw \
--class web \
--service web_common \
--evtattr event_name:"Web event1" \
--receiver res:"drive.google.com",res_destination_url:"/file/test" \
--data file:/root/test
```

In the command above, `test` is a file that contains a POST query. In our example, the query contains the *strictly confidential* phrase. When you configure data protection policies, the event looks as follows:



The `data` and `receiver` attributes are required.

Example on how to send events of *Photo* class

To send a *Photo* class event to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token your_plugin_token_value \
--id iw \
--class photo \
--service photosof \
--evtattr event_name:"Photo event" \
--receiver ws:"ws1.domain.org",workstation_type:"ws_type_computer" \
--data file:/root/test.png,filename:test.png
```

Examples on how to send events of *File Exchange* class

The `receiver`, `data` and `destination file path` attributes are required for events of `Cloud storage`, `Terminal session`, `Network resourse`, `FTP` and `External device` services.

The `receiver` and `data` attributes are required for *File Exchange* class events of `Messengers` service.

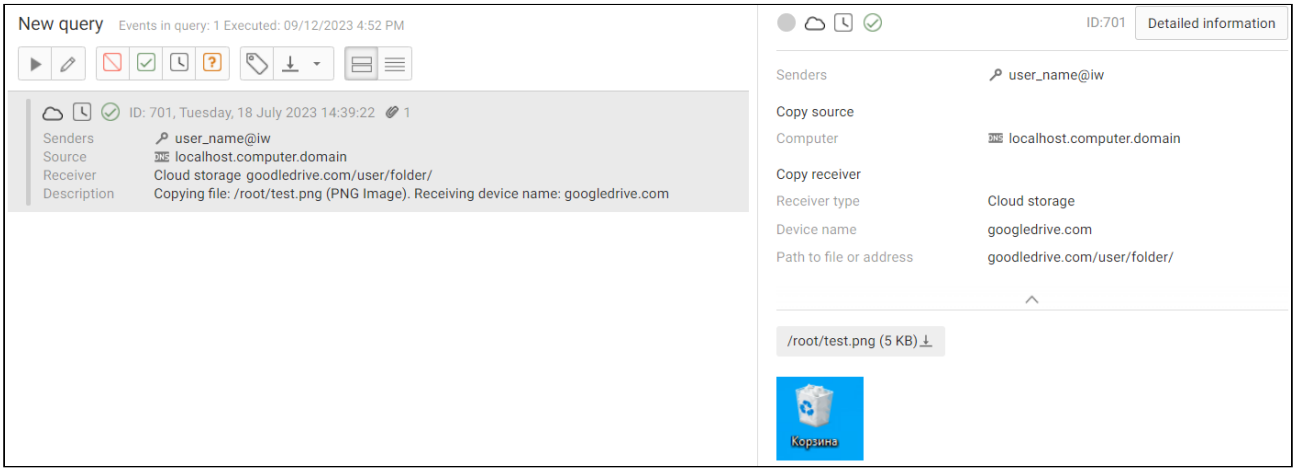
Example 1:

To send *File Exchange* class event of `Cloud Storage` service, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class fileexch \
--service cloud_storage \
--evtattr event_name:"cloud_storage event" \
--sender auth:user_name@iw \
```

```
--receiver res:googledrive.com,res_destination_url:"goodledrive.com/user/folder/"  \
--data file:/root/test.png,destination_file_path:"Send file here"
```

In Traffic Monitor Management console, the event looks as follows:



Example 2:

To send a *File exchange* class event of `Terminal session` service to Traffic Monitor, use the command below:

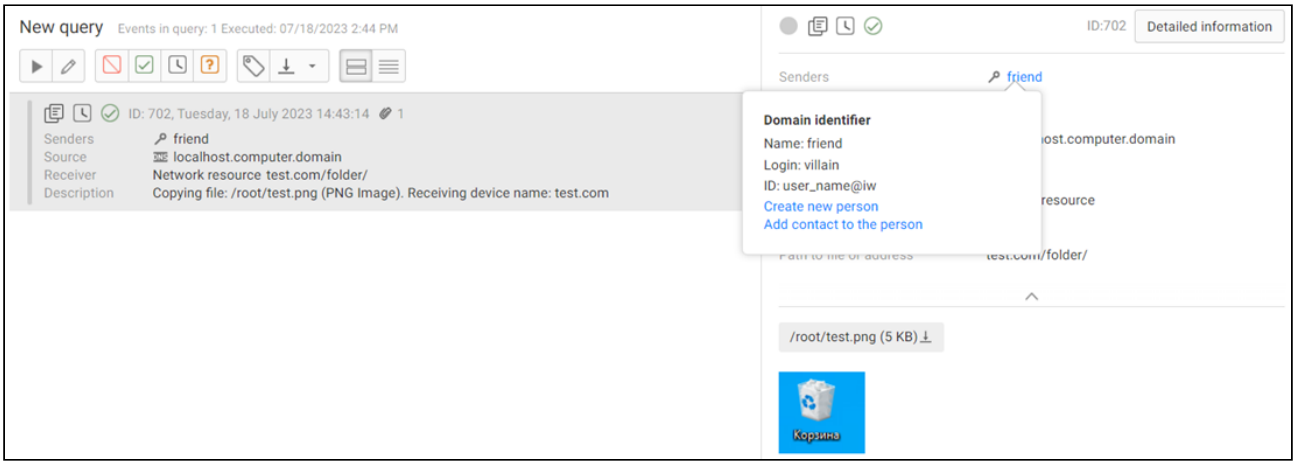
```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class fileexch \
--service terminal_session \
--evtattr event_name:"terminal_session event" \
--sender phone:+79057777777 \
--receiver res:root@test.com \
--data file:/root/test.png,destination_file_path:"test.com/folder"
```

Example 3:

To send a *File exchange* class event of `Network resources` service to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class fileexch \
--service network_resource \
--evtattr event_name:"network_resource event" \
--sender "auth:user_name@iw:'{\"display_name\":\"friend\",\"login\":\"villain\"}'" \
--receiver res:test.com,res_destination_url:"test.com/folder/ "  \
--data file:/root/test.png,destination_file_path:"Send file here"
```

In Traffic Monitor Management console, the event looks as follows:



Example 4:

To send a *File exchange* class event of `FTP` service to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class fileexch \
--service ftp \
--evtattr event_name:"FTP Event" \
--sender login:villain \
--receiver res:ftp.site.org,res_destination_url:"catalog\catalog\res.file" \
--data file:/root/test.png,destination_file_path:"Send file here"
```

Example 5:

To send a *File exchange* class event of `External device` service to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class fileexch \
--service file_copy_removable \
--evtattr event_name:"Copy file on device" \
--receiver dev:"Removable device" \
--data file:/root/test.png,source_file_path:"Copy from here",destination_file_path:"Send file here"
```

Example 6:

To send a *File exchange* class event for ICQ application to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class fileexch \
--service im_icq \
--evtattr event_name:"ICQ File Transfer Event" \
```

```
--receiver icq:IcqReceiver \
--data file:/root/test.png
```

Example 7:

To send a *File exchange* class event for Skype application to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class fileexch \
--service im_skype \
--evtattr event_name:"Skype File Transfer Event" \
--sender skype:SkypeSender \
--receiver skype:SkypeReceiver1 \
--receiver skype:SkypeReceiver2 \
--data file:/root/test.png,filename:test.png
```

Example on how to send events of *Call* class

To send a *Call* class event of `Messengers` service to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token device_monitor_token_value \
--id iw \
--class voicetalk \
--service im_skype \
--evtattr event_name:"Skype Voice Talk" \
--sender skype:SkypeSender \
--receiver skype:SkypeReceiver \
--data file:/root/sound_msg.wav,call_duration:7
```

The event contains `sound_msg.wav` audio file with the call duration of 7 seconds. The file is available for download in Traffic Monitor Management Console. The `receiver`, `data` and `call_duration` attributes are required for *Call* event classes of `Messengers` service.

Examples on how to send events of *Recording media* class

To send a *Recording media* class event to Traffic Monitor, use the command below:

```
pushapi-util \
--host TM_host \
--token your_plugin_token_value \
--id iw \
--class multimedia \
--service videosof \
--evtattr event_name:"Video event" \
--receiver ws:"ws1.domain.org",workstation_type:"ws_type_computer" \
--data file:/root/video.avi
```

Building samples under Linux in PushAPI SDK

To build samples under Linux, you need the installed **cmake** package and the compiler supporting C++11 standard. All other dependencies are defined by **cmake** script. To build the samples, you may need to install Thrift Runtime libraries, Boost library, and OpenSSL library, namely the following ones:

- boost 1.62;
- openssl 1.0.1;
- thrift 0.11.0.

Building samples under Windows in PushAPI SDK

To build samples under Windows, you need the installed Visual Studio 2013 program. The samples include Toolchain that contains boost, OpenSSL, and source codes of Thrift Runtime libraries which are built as a part of the sample. Toolchain includes a thrift utility that compiles the thrift scheme to a corresponding source code. It is possible to use custom implementation of boost, OpenSSL, and Thrift – for these the user should define the environment variables:

```
$(BOOST_ROOT)
$(OPENSSL_ROOT_DIR)
$(THRIFT_INSTALL_DIR)
```

11.1.5 Troubleshooting When Sending Events

If you experience an issue when sending events via PushAPI, try looking into:

- Web interface problems. To activate the debug mode on the server:
  1. Connect to the server using SSH.
  2. Open the `/opt/iw/tm5/etc/web.conf` file for editing.
  3. Enable the debug mode by setting the `"debug": true` value.
  4. Save the file.
  5. Refresh the web page.
  6. In your browser search box, go to the debug console to see detailed logs on requests sent to the API: `https://<server_IP>/api/debug/default`.
- Missing trusted certificates. Add self-signed certificates as trusted certificates. Paths to certificates:
  - Web: `/opt/iw/tm5/etc/certification/web-server.pem`;
  - XAPI: `/opt/iw/tm5/etc/cert/server.pem`;
- File queues. To see all file and error queues on the Traffic Monitor server, go to `opt/iw/tm5/queue` and `opt/iw/tm5/errors` respectively.
- Files with extraction errors. In the `/opt/iw/tm5/etc/extractors.conf` configuration file, there is a `Dreamcatcher` parameter for each extractor. The folder with the files is specified in the parameter.
- Operation of the iw_xapi_xapi and iw_xapi_puppy services. To get information from different program component loggers:
  1. Go to the Traffic Monitor server and open the `/opt/iw/tm5/etc/xapi.conf` configuration file for editing.
  2. Set the logger values:

```
"GlobalLevel": "debug",
"Loggers":
{
"Filequeue": "debug",
"RawMailDump": "fatal",
"Root": "debug",
"puppy": "debug",
"thrift_handler": "warning"
}
```

In the code above:

- `Filequeue` – file queue logs;
- `RawMailDump` – raw mail data;
- `puppy` – PushAPI logger;
- `thrift_handler` – data on thrift structures, which are received via the network. Used for traffic check;
- `Root` – root logger designed for reading from a database, index creation, archiving and index removal.

3. Save the file.
4. Run the commands:

```
iwtm restsrt xapi_xapi
iwtm restart xapi_puppy
```

5. Send the event to Traffic Monitor via Push API. Log files are saved to the `/var/log/infowatch` directory. See *Examples on How to Use the pushapi-util Utility* for details.

11.2 Uploading Events via HTTP Protocol

To upload events to Traffic Monitor via HTTP protocol, use `JsonPush` REST API method, which transforms a query into PushAPI format. For details on REST API, see *Functional description of REST API programming interface*.

> **Warning!**
>
> Make sure that `iw_rest2push` service is configured and running. For details, see *InfoWatch Traffic Monitor. Configuration Files Guide. rest2push.conf and iw_rest2push.service*.

API Version: ≥ 1.8

Method: `POST`

Resource: `/push/event`

Description: The method sends the event to Traffic Monitor via HTTP protocol.

Query examples are provided in the articles:

- Uploading Events with Web Message Type;
- Uploading Events with Messenger Type;
- Uploading Events with File Exchange Type;
- Uploading Events with Print Type.

Successful response example:

```
{
  "data": {
    "document_id": "string"
  }
}
```

There are several ways to get verdict for an event sent to Traffic Monitor:

- via web api using the 443 port. This way is available in any supported Traffic Monitor version (for details, see *Getting verdict*).
- using **rest2push** service using the 9106 port. From the information security point of view, this way is preferable, as the connection is performed directly to the **rest2push** service instance, without using Traffic Monitor web interface. There is also a possibility to connect to any of deployed **rest2push** services to distribute the load. It is available starting from Traffic Monitor version 7.11. See the request example below.
- via **rest2push** using web socket (see the details in the article *Uploading Events to Get Traffic Monitor Verdict*). Available starting from Traffic Monitor version 7.13.

Getting verdict for events uploaded via HTTP protocol

To get verdict for an event, use DataExport API. Provide an array of document ID, which you received as output after the event had been uploaded to Traffic Monitor, and date: `{documentID, date}`.

Example of getting verdict for two events:

```
curl -v -k -H "X-API-CompanyId: iw" \
-H "X-API-ProductName: PRODUCT" \
-H "X-API-Auth-Token: akkw713dh9a1awwhrn1" \
-H "X-API-Version: 1.8" \
"https://server.company.com:9106/xapi/event/verdict?documentId=41a43cf1-bec3-44ae-9fd9-
c1fe962e7886,7b6ad1fe-413c-486e-9dd1-c8e5a94a198b&date=2024-07-17"
```

In the example above, `server.company.com` is a server with *Interceptors* role.

For details, see *Getting verdict*.

Recommendations on API performance when uploading events to TM:

- To improve NGINX performance:
  1. In `/etc/nginx/nginx.conf`, change the `worker_processes` parameter value to 2.
  2. Restart the service with the command:

     ```
     systemctl restart nginx
     ```

- In case of high load, the NGINX log file `/var/log/nginx/error.log` may contain messages which look as follows:

  ```
  socket() failed (24: Too many open files) while connecting to upstream.
  ```

  To avoid the error, increase the file-opening limit of the service, for example, to 10000:

  a. Enter the NGINX service editing mode using the command:

```
systemctl edit nginx.service
```

b. Paste the following lines into the editor:

```
[Service]

LimitNOFILE=10000
```

c. Save changes (**Ctrl**+**O**) and exit (**Ctrl**+**X**).
d. Restart the service with the command:

```
systemctl restart nginx
```

11.2.1 Uploading Events with Web Message Type

Query example:

```
curl -X 'POST' \
'https://server.company.com/xapi/push/event' \
-H 'accept: application/json' \
-H 'X-API-Auth-Token: 198igrnlnuonix672uq9' \
-H 'X-API-CompanyId: Yandex' \
-H 'X-API-Version: 1.8' \
-H 'Content-Type: multipart/form-data' \
-H 'X-Enable-OpenAPI: 1' \
-F 'event=
{
  "evt_class": 3,
  "evt_service": "web_common",
  "evt_attributes": [
      { "name": "event_name", "value": "Web Event Name" },
      { "name": "capture_ts", "value": "2024-03-25T23:41:21+03:00" },
      { "name": "capture_server_ip", "value": "127.0.0.1" },
      { "name": "capture_server_fqdn", "value": "captureservername.domain" }
  ],
  "evt_senders": [
      {
          "identity_id": 3,
          "identity_type": 0,
          "identity_contacts": [],
          "identity_contacts_with_meta": [
              {
              "contact": { "name": "auth", "value": "userlogin@domain" }
              }
          ]
      },
      {
          "identity_id": 4,
          "identity_type": 1,
          "identity_contacts": [],
          "identity_attributes": [
              { "name": "workstation_type", "value": "ws_type_computer" }
```

```
            ],
            "identity_contacts_with_meta": [
                {
                "contact": { "name": "dnshostname", "value": "computername.domain" }
                }
            ]
        }
    ],
    "evt_receivers": [
        {
            "identity_id": 2,
            "identity_type": 3,
            "identity_contacts": [],
            "identity_attributes": [
                { "name": "res_destination_host", "value": "testhost.com" },
                { "name": "res_destination_url", "value": "\/file\/test.php" }
            ]
        }
    ],
    "evt_data": [
        {
            "data_id": 1
        }
    ]
}
```

Note:

In the example above, value `3` of the `"evt_class"` field corresponds to the `kWeb` constant. Along with it, you must send a POST request, for example, in *multipart* format.

Example query for text events being sent

```
POST /file/test.php HTTP/1.1
Content-Type: multipart/form-data; boundary="----1234567890----"
Content-Length: 110
Host: testhost.com

------1234567890----
Content-Type: text/plain
Content-Disposition: inline

Hello World
------1234567890------
```

Example query for files being sent

```
POST /file/test.php HTTP/1.1
Content-Type: multipart/form-data; boundary="----1234567890----"
Content-Length: 156
Host: testhost.com

------1234567890----
```

```
Content-Type: text/plain
Content-Disposition: attachment; filename="file.txt"

Hello World.
This is text file.

------1234567890------
```

11.2.2 Uploading Events with Messenger Type

The chapter contains examples of queries for uploading *Messenger* type events. The examples use Telegram messenger.

- Uploading Calls;
- Uploading Text Messages;
- Uploading Files.

Uploading Calls

Query example:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
  -H 'X-Enable-OpenAPI: 1' \
  -F 'event={
  "evt_class": 7,
  "evt_service": "im_telegram",
  "evt_attributes": [
    {
      "name": "event_name",
      "value": "Telegram call"
    },
    {
      "name": "capture_ts",
      "value": "2025-06-02T14:36:52+03:00"
    },
    {
      "name": "capture_server_ip",
      "value": "127.0.0.1"
    },
    {
      "name": "capture_server_fqdn",
      "value": "localhost"
    }
  ],
  "evt_senders": [
    {
      "identity_id": 3,
      "identity_type": 0,
```

```
      "identity_contacts": [],
      "identity_contacts": [
        {
            "name": "Telegram",
            "value": "TelegramSender"
        }
      ]
    },
    {
      "identity_id": 4,
      "identity_type": 1,
      "identity_contacts": [],
      "identity_attributes": [
        {
          "name": "workstation_type",
          "value": "ws_type_computer"
        }
      ],
      "identity_contacts": [
        {
            "name": "dnshostname",
            "value": "localhost.computer.domain"
        }
      ]
    }
  ],
  "evt_receivers": [
    {
      "identity_id": 2,
      "identity_type": 0,
      "identity_contacts": [],
      "identity_contacts": [
        {
            "name": "telegram",
            "value": "TelegramReceiver"
        }
      ]
    }
  ],
  "evt_data": [
    {
      "data_id": 1,
      "data_attributes": [
        {
          "name": "call_duration",
          "value": "18"
        },
        {
          "name": "sent_date",
          "value": "2025-06-02T14:36:52+03:00"
        }
      ]
    }
  ]
}' \
  -F 'files[]='
```

Uploading Text Messages

Query example:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
  -H 'X-Enable-OpenAPI: 1' \
  -F 'event={
  "evt_class": 0,
  "evt_service": "im_telegram",
  "evt_attributes": [
    {
      "name": "event_name",
      "value": "Text message"
    },
    {
      "name": "capture_ts",
      "value": "2025-06-02T13:52:10+03:00"
    },
    {
      "name": "capture_server_ip",
      "value": "127.0.0.1"
    },
    {
      "name": "capture_server_fqdn",
      "value": "localhost"
    }
  ],
  "evt_senders": [
    {
      "identity_id": 2,
      "identity_type": 0,
      "identity_contacts": [],
      "identity_contacts": [
        {
            "name": "im_telegram",
            "value": "TelegramSender"
        }
      ]
    },
    {
      "identity_id": 4,
      "identity_type": 0,
      "identity_contacts": [],
      "identity_contacts": [
        {
            "name": "im_telegram",
            "value": "TelegramReceiver1"
        }
      ]
    },
    {
```

```
    "identity_id": 7,
    "identity_type": 0,
    "identity_contacts": [],
    "identity_contacts": [
      {
          "name": "telegram",
          "value": "TelegramSender"
      }
    ]
  },
  {
    "identity_id": 8,
    "identity_type": 1,
    "identity_contacts": [],
    "identity_attributes": [
      {
        "name": "workstation_type",
        "value": "ws_type_computer"
      }
    ],
    "identity_contacts": [
      {
          "name": "dnshostname",
          "value": "localhost.computer.domain"
      }
    ]
  }
],
"evt_receivers": [
  {
    "identity_id": 1,
    "identity_type": 0,
    "identity_contacts": [],
    "identity_contacts": [
      {
          "name": "im_telegram",
          "value": "TelegramSender"
      }
    ]
  },
  {
    "identity_id": 3,
    "identity_type": 0,
    "identity_contacts": [],
    "identity_contacts": [
      {
          "name": "im_telegram",
          "value": "TelegramReceiver1"
      }
    ]
  },
  {
    "identity_id": 5,
    "identity_type": 0,
    "identity_contacts": [],
    "identity_contacts": [
      {
          "name": "telegram",
          "value": "TelegramReceiver1"
```

```
        }
      ]
    },
    {
      "identity_id": 6,
      "identity_type": 0,
      "identity_contacts": [],
      "identity_contacts": [
        {
            "name": "telegram",
            "value": "TelegramSender"
        }
      ]
    }
  ],
  "evt_data": [],
  "evt_messages": [
    {
      "sender_id": 2,
      "sent_time": "2025-06-02T13:52:12+03:00",
      "utf8_text": "Strictly confidential"
    },
    {
      "sender_id": 4,
      "sent_time": "2025-06-02T13:52:13+03:00",
      "utf8_text": "Thanks for info"
    }
  ]
}' \
  -F 'files[]='
```

Uploading Files

Query example:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
  -H 'X-Enable-OpenAPI: 1' \
  -F 'event={
  "evt_class": 4,
  "evt_service": "im_telegram",
  "evt_attributes": [
    {
      "name": "event_name",
      "value": "Telegram File Transfer Event"
    },
    {
      "name": "capture_ts",
      "value": "2025-06-02T14:25:53+03:00"
    },
    {
```

```
    "name": "capture_server_ip",
    "value": "127.0.0.1"
  },
  {
    "name": "capture_server_fqdn",
    "value": "localhost"
  }
],
"evt_senders": [
  {
    "identity_id": 3,
    "identity_type": 0,
    "identity_contacts": [],
    "identity_contacts": [
      {
          "name": "telegram",
          "value": "TelegramSender"
      }
    ]
  },
  {
    "identity_id": 4,
    "identity_type": 1,
    "identity_contacts": [],
    "identity_attributes": [
      {
        "name": "workstation_type",
        "value": "ws_type_computer"
      }
    ],
    "identity_contacts": [
      {
          "name": "dnshostname",
          "value": "localhost.computer.domain"
      }
    ]
  }
],
"evt_receivers": [
  {
    "identity_id": 2,
    "identity_type": 0,
    "identity_contacts": [],
    "identity_contacts": [
      {
          "name": "telegram",
          "value": "TelegramReceiver1"
      }
    ]
  }
],
"evt_data": [
  {
    "data_id": 1,
    "data_attributes": [
      {
        "name": "filename",
        "value": "/root/test.txt"
      },
```

```
        {
          "name": "sent_date",
          "value": "2025-06-02T14:25:53+03:00"
        }
      ]
    }
  ]
}' \
  -F 'files[]='
```

11.2.3 Uploading Events with File Exchange Type

The section contains query examples for uploading events with *File exchange* type.

- Query for a File copied to a Network Resource;
- Query for a File copied to a Removable Device;
- Query for a File copied via Terminal Session;
- Query for a File sent to a Cloud Storage;
- Query for a File sent via FTP.

Copying File to Network Resource

Query example for uploading a single file:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
  -H 'X-Enable-OpenAPI: 1' \
  -F 'event={
  "evt_class": 4,
  "evt_service": "network_resource",
  "evt_attributes": [
    {
      "name": "event_name",
      "value": "network_resource event"
    },
    {
      "name": "capture_ts",
      "value": "2025-05-28T13:19:41+03:00"
    },
    {
      "name": "capture_server_ip",
      "value": "127.0.0.1"
    },
    {
      "name": "capture_server_fqdn",
      "value": "localhost"
    }
  ],
  "evt_senders": [
    {
```

```
      "identity_id": 3,
      "identity_type": 0,
      "identity_contacts": [
        {
          "name": "auth",
          "value": "user_name@iw"
        }
      ]
    },
    {
      "identity_id": 4,
      "identity_type": 1,
      "identity_contacts": [
        {
          "name": "dnshostname",
          "value": "localhost.computer.domain"
        }
      ],
      "identity_attributes": [
        {
          "name": "workstation_type",
          "value": "ws_type_computer"
        }
      ]
    }
  ],
  "evt_receivers": [
    {
      "identity_id": 2,
      "identity_type": 3,
      "identity_contacts": [],
      "identity_attributes": [
        {
          "name": "res_destination_host",
          "value": "test.com"
        },
        {
          "name": "res_destination_url",
          "value": "test.com/folder/ "
        }
      ]
    }
  ],
  "evt_data": [
    {
      "data_id": 2,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test.png"
        },
        {
          "name": "destination_file_path",
          "value": "Send file here"
        }
      ]
    }
  ]
}' \
```

```
  -F 'files[]=@2'
```

Query example for uploading multiple files:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
  -H 'X-Enable-OpenAPI: 1' \
  -F 'event={
  "evt_class": 4,
  "evt_service": "network_resource",
  "evt_attributes": [
    {
      "name": "event_name",
      "value": "network_resource event"
    },
    {
      "name": "capture_ts",
      "value": "2025-06-03T16:17:41+03:00"
    },
    {
      "name": "capture_server_ip",
      "value": "127.0.0.1"
    },
    {
      "name": "capture_server_fqdn",
      "value": "localhost"
    }
  ],
  "evt_senders": [
    {
      "identity_id": 3,
      "identity_type": 0,
      "identity_contacts": [
        {
          "name": "auth",
          "value": "user_name@iw"
        }
      ]
    },
    {
      "identity_id": 4,
      "identity_type": 1,
      "identity_contacts": [
        {
          "name": "dnshostname",
          "value": "localhost.computer.domain"
        }
      ],
      "identity_attributes": [
        {
          "name": "workstation_type",
          "value": "ws_type_computer"
```

```
        }
      ]
    }
  ],
  "evt_receivers": [
    {
      "identity_id": 2,
      "identity_type": 3,
      "identity_contacts": [],
      "identity_attributes": [
        {
          "name": "res_destination_host",
          "value": "test.com"
        },
        {
          "name": "res_destination_url",
          "value": "test.com/folder/ "
        }
      ]
    }
  ],
  "evt_data": [
    {
      "data_id": 0,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test.png"
        },
        {
          "name": "destination_file_path",
          "value": "Send file here"
        }
      ]
    },
    {
      "data_id": 1,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test1.png"
        },
        {
          "name": "destination_file_path",
          "value": "Send1 file here"
        }
      ]
    }
  ]
}
' \
  -F 'files[]=@/root/test.png;filename=0' \
  -F 'files[]=@/root/test1.png;filename=1'
```

Note:

In `-F` node, `filename` parameter value corresponds to `data_id` field value.

Copying File to Removable Device

Query example for uploading a single file:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
  -H 'X-Enable-OpenAPI: 1' \
  -F 'event={
  "evt_class": 4,
  "evt_service": "file_copy_removable",
  "evt_attributes": [
    {
      "name": "event_name",
      "value": "Copy file on device"
    },
    {
      "name": "capture_ts",
      "value": "2025-05-28T12:07:42+03:00"
    },
    {
      "name": "capture_server_ip",
      "value": "127.0.0.1"
    },
    {
      "name": "capture_server_fqdn",
      "value": "localhost"
    }
  ],
  "evt_senders": [
    {
      "identity_id": 3,
      "identity_type": 0,
      "identity_contacts": [
        {
          "name": "auth",
          "value": "root@localhost"
        }
      ]
    },
    {
      "identity_id": 4,
      "identity_type": 1,
      "identity_contacts": [
        {
          "name": "dnshostname",
          "value": "localhost.computer.domain"
        }
      ],
```

```
      "identity_attributes": [
        {
          "name": "workstation_type",
          "value": "ws_type_computer"
        }
      ]
    }
  ],
  "evt_receivers": [
    {
      "identity_id": 2,
      "identity_type": 2,
      "identity_contacts": [],
      "identity_attributes": [
        {
          "name": "device_name",
          "value": "Removable device"
        }
      ]
    }
  ],
  "evt_data": [
    {
      "data_id": 2,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test.png"
        },
        {
          "name": "source_file_path",
          "value": "Copy from here"
        },
        {
          "name": "destination_file_path",
          "value": "Send file here"
        }
      ]
    }
  ]
}' \
  -F 'files[]=@2'
```

Query example for uploading multiple files:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
  -H 'X-Enable-OpenAPI: 1' \
  -F 'event={
  "evt_class": 4,
  "evt_service": "file_copy_removable",
```

```
"evt_attributes": [
  {
    "name": "event_name",
    "value": "Copy file on device"
  },
  {
    "name": "capture_ts",
    "value": "2025-06-03T16:27:12+03:00"
  },
  {
    "name": "capture_server_ip",
    "value": "127.0.0.1"
  },
  {
    "name": "capture_server_fqdn",
    "value": "localhost"
  }
],
"evt_senders": [
  {
    "identity_id": 3,
    "identity_type": 0,
    "identity_contacts": [
      {
        "name": "auth",
        "value": "root@localhost"
      }
    ]
  },
  {
    "identity_id": 4,
    "identity_type": 1,
    "identity_contacts": [
      {
        "name": "dnshostname",
        "value": "localhost.computer.domain"
      }
    ],
    "identity_attributes": [
      {
        "name": "workstation_type",
        "value": "ws_type_computer"
      }
    ]
  }
],
"evt_receivers": [
  {
    "identity_id": 2,
    "identity_type": 2,
    "identity_contacts": [],
    "identity_attributes": [
      {
        "name": "device_name",
        "value": "Removable device"
      }
    ]
  }
],
```

```
  "evt_data": [
    {
      "data_id": 0,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test.png"
        },
        {
          "name": "source_file_path",
          "value": "Copy from here"
        },
        {
          "name": "destination_file_path",
          "value": "Send file here"
        }
      ]
    },
    {
      "data_id": 1,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test1.png"
        },
        {
          "name": "source_file_path",
          "value": "Copy from here"
        },
        {
          "name": "destination_file_path",
          "value": "Send file here"
        }
      ]
    }
  ]
}' \
  -F 'files[]=@/root/test.png;filename=0' \
  -F 'files[]=@/root/test1.png;filename=1'
```

Note:

In `-F` node, `filename` parameter value corresponds to `data_id` field value.

Copying File via Terminal Session

Query example for uploading a single file:

```
curl -X 'POST' \
  'https://server.company.ru/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
```

```
-H 'Content-Type: multipart/form-data' \
-H 'X-Enable-OpenAPI: 1' \
-F 'event={
"evt_class": 4,
"evt_service": "terminal_session",
"evt_attributes": [
  {
    "name": "event_name",
    "value": "terminal_session event"
  },
  {
    "name": "capture_ts",
    "value": "2025-05-28T09:09:44+03:00"
  },
  {
    "name": "capture_server_ip",
    "value": "127.0.0.1"
  },
  {
    "name": "capture_server_fqdn",
    "value": "localhost"
  }
],
"evt_senders": [
  {
    "identity_id": 3,
    "identity_type": 0,
    "identity_contacts": [
      {
        "name": "phone",
        "value": "+79123456789"
      }
    ]
  },
  {
    "identity_id": 4,
    "identity_type": 1,
    "identity_contacts": [
      {
        "name": "dnshostname",
        "value": "localhost.computer.domain"
      }
    ],
    "identity_attributes": [
      {
        "name": "workstation_type",
        "value": "ws_type_computer"
      }
    ]
  }
],
"evt_receivers": [
  {
    "identity_id": 2,
    "identity_type": 3,
    "identity_contacts": [],
    "identity_attributes": [
      {
        "name": "res_destination_host",
```

```
          "value": "root@test.com"
        }
      ]
    }
  ],
  "evt_data": [
    {
      "data_id": 1,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test.png"
        },
        {
          "name": "destination_file_path",
          "value": "test.com/folder"
        }
      ]
    }
  ]
}' \
  -F 'files[]=@/root/test.png;filename=1'
```

Query example for uploading multiple files:

```
curl --insecure -X 'POST' \
  'https://server.company.com/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
  -H 'X-Enable-OpenAPI: 1' \
  -F 'event={
  "evt_class": 4,
  "evt_service": "terminal_session",
  "evt_attributes": [
    {
      "name": "event_name",
      "value": "terminal_session event"
    },
    {
      "name": "capture_ts",
      "value": "2025-06-03T16:35:14+03:00"
    },
    {
      "name": "capture_server_ip",
      "value": "127.0.0.1"
    },
    {
      "name": "capture_server_fqdn",
      "value": "localhost"
    }
  ],
  "evt_senders": [
    {
      "identity_id": 3,
```

```
      "identity_type": 0,
      "identity_contacts": [
        {
          "name": "phone",
          "value": "+79123456789"
        }
      ]
    },
    {
      "identity_id": 4,
      "identity_type": 1,
      "identity_contacts": [
        {
          "name": "dnshostname",
          "value": "localhost.computer.domain"
        }
      ],
      "identity_attributes": [
        {
          "name": "workstation_type",
          "value": "ws_type_computer"
        }
      ]
    }
  ],
  "evt_receivers": [
    {
      "identity_id": 2,
      "identity_type": 3,
      "identity_contacts": [],
      "identity_attributes": [
        {
          "name": "res_destination_host",
          "value": "root@test.com"
        }
      ]
    }
  ],
  "evt_data": [
    {
      "data_id": 0,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test.png"
        },
        {
          "name": "destination_file_path",
          "value": "test.com/folder"
        }
      ]
    },
    {
      "data_id": 1,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test1.png"
        },
```

```
        {
          "name": "destination_file_path",
          "value": "test.com/folder"
        }
      ]
    }
  ]
}' \
  -F 'files[]=@/root/test.png;filename=0' \
  -F 'files[]=@/root/test1.png;filename=1'
```

> **Note:**
>
> In `-F` node, `filename` parameter value corresponds to `data_id` field value.

Sending File to Cloud Storage

Query example:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
   'X-API-Auth-Token: brse99dugp1cdbr69axz'
   'X-API-CompanyId: iw'
   'X-API-Version: 1.8'
  -F 'event={
  "evt_class": 4,
  "evt_service": "cloud_storage",
  "evt_attributes": [
    {
      "name": "capture_ts",
      "value": "2023-10-26T20:07:51+03:00"
    },
    {
      "name": "capture_server_ip",
      "value": "127.0.0.1"
    },
    {
      "name": "capture_server_fqdn",
      "value": "localhost"
    }
  ],
  "evt_senders": [
    {
      "identity_id": 1,
      "identity_type": 0,
      "identity_contacts": [
        {
          "name": "auth",
          "value": "user_name@iw"
        }
      ]
    },
    {
      "identity_id": 2,
```

```
      "identity_type": 1,
      "identity_contacts": [
        {
          "name": "dnshostname",
          "value": "localhost.computer.domain"
        }
      ]
    }
  ],
  "evt_receivers": [
    {
      "identity_id": 3,
      "identity_type": 3,
      "identity_contacts": []
    }
  ],
  "evt_data": [
    {
      "data_id": 1,
      "data_attributes": [
        {
          "name": "filename",
          "value": "res.txt"
        },
        {
          "name": "destination_file_path",
          "value": "googledrive.com/user/folder/"
        }
      ]
    }
  ],
  "evt_destination": {
    "identity_id": 4,
    "identity_type": 3,
    "identity_contacts": [],
    "identity_attributes": [
      {
        "name": "name",
        "value": "googledrive.com"
      }
    ]
  }
}' \
  -F 'files[]='
```

Sending File via FTP

Query example for uploading a single file:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
```

```
-H 'X-Enable-OpenAPI: 1' \
-F 'event={
"evt_class": 4,
"evt_service": "ftp",
"evt_attributes": [
  {
    "name": "event_name",
    "value": "FTP Event"
  },
  {
    "name": "capture_ts",
    "value": "2025-05-27T15:49:56+03:00"
  },
  {
    "name": "capture_server_ip",
    "value": "127.0.0.1"
  },
  {
    "name": "capture_server_fqdn",
    "value": "localhost"
  }
],
"evt_senders": [
  {
    "identity_id": 3,
    "identity_type": 0,
    "identity_contacts": [
      {
        "name": "login",
        "value": "villain"
      }
    ]
  },
  {
    "identity_id": 4,
    "identity_type": 1,
    "identity_contacts": [
      {
        "name": "dnshostname",
        "value": "localhost.computer.domain"
      }
    ],
    "identity_attributes": [
      {
        "name": "workstation_type",
        "value": "ws_type_computer"
      }
    ]
  }
],
"evt_receivers": [
  {
    "identity_id": 2,
    "identity_type": 3,
    "identity_contacts": [],
    "identity_attributes": [
      {
        "name": "res_destination_host",
        "value": "ftp.site.org"
```

```
        },
        {
          "name": "res_destination_url",
          "value": "catalog\\catalog\\res.file"
        }
      ]
    }
  ],
  "evt_data": [
    {
      "data_id": 1,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test.png"
        },
        {
          "name": "destination_file_path",
          "value": "Send file here"
        }
      ]
    }
  ]
}' \
  -F 'files[]=@/root/test.png;filename=1'
```

Query example for uploading multiple files:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: brse99dugp1cdbr69axz' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
  -H 'X-Enable-OpenAPI: 1' \
  -F 'event={
  "evt_class": 4,
  "evt_service": "ftp",
  "evt_attributes": [
    {
      "name": "event_name",
      "value": "FTP Event"
    },
    {
      "name": "capture_ts",
      "value": "2025-06-03T16:03:10+03:00"
    },
    {
      "name": "capture_server_ip",
      "value": "127.0.0.1"
    },
    {
      "name": "capture_server_fqdn",
      "value": "localhost"
    }
  ],
```

```
"evt_senders": [
  {
    "identity_id": 3,
    "identity_type": 0,
    "identity_contacts": [
      {
        "name": "login",
        "value": "villain"
      }
    ]
  },
  {
    "identity_id": 4,
    "identity_type": 1,
    "identity_contacts": [
      {
        "name": "dnshostname",
        "value": "localhost.computer.domain"
      }
    ],
    "identity_attributes": [
      {
        "name": "workstation_type",
        "value": "ws_type_computer"
      }
    ]
  }
],
"evt_receivers": [
  {
    "identity_id": 2,
    "identity_type": 3,
    "identity_contacts": [],
    "identity_attributes": [
      {
        "name": "res_destination_host",
        "value": "ftp.site.org"
      },
      {
        "name": "res_destination_url",
        "value": "catalog\\catalog\\res.file"
      }
    ]
  }
],
"evt_data": [
  {
    "data_id": 0,
    "data_attributes": [
      {
        "name": "filename",
        "value": "/root/test.png"
      },
      {
        "name": "destination_file_path",
        "value": "Send file here"
      }
    ]
  },
```

```
    {
      "data_id": 1,
      "data_attributes": [
        {
          "name": "filename",
          "value": "/root/test1.png"
        },
        {
          "name": "destination_file_path",
          "value": "Send file here"
        }
      ]
    }
  ]
}' \
  -F 'files[]=@/root/test.png;filename=0' \
  -F 'files[]=@/root/test1.png;filename=1'
```

> **Note:**
>
> In `-F` node, `filename` parameter value corresponds to `data_id` field value.

11.2.4 Uploading Events with Print Type

Query example:

```
curl -X 'POST' \
  'https://server.company.com/xapi/push/event' \
   'X-API-Auth-Token: brse99dugp1cdbr69axz'
   'X-API-CompanyId: iw'
   'X-API-Version: 1.8'
  -F 'event=
{
  "evt_class": 2,
  "evt_service": "print",
  "evt_attributes": [
    {"name": "capture_ts", "value": "2024-03-06T08:45:00+03:00"},
    {"name": "capture_server_ip", "value": "127.0.0.1"},
    {"name": "capture_server_fqdn", "value": "captureservername.domain"},
    {"name": "print_copies", "value": "3"}
  ],
  "evt_senders": [
    {
      "identity_id": 3,
      "identity_type": 0,
      "identity_contacts": [
        {"name": "auth", "value": "userlogin@domain"},
        {"name": "email", "value": "test@example.com"}
      ]
    },
    {
      "identity_id": 4,
      "identity_type": 1,
      "identity_contacts": [
```

```
        {"name": "dnshostname", "value": "computername.domain"}
      ],
      "identity_attributes": [
        {"name": "workstation_type", "value": "ws_type_computer"}
      ]
    }
  ],
  "evt_receivers": [
    {
      "identity_id": 2,
      "identity_type": 2,
      "identity_contacts": [],
      "identity_attributes": [
        {"name": "device_name", "value": "PrinterName"},
        {"name": "print_port_name", "value": "Some print port name"},
        {"name": "print_server", "value": "Printer Xerox"},
        {"name": "print_location", "value": "Kitchen"}
      ]
    }
  ],
  "evt_data": [
    {
      "data_id": 1,
      "data_attributes": [
        {"name": "print_copies", "value": "3"},
        {"name": "print_job_name", "value": "Print job name"}
      ]
    }
  ]
}' \
  -F 'files[]='
```

11.2.5 Uploading Events to Get Traffic Monitor Verdict

External systems can connect to Traffic Monitor, send events for analysis to it, and get a real-time verdict via a web socket.

To set up sending events and getting responses:

1. To open a web socket, send a command in the terminal on the client side (on an external system), specifying the valid IP address of the Traffic Monitor server:

   ```
   wscat -c 'wss://<ip_adress>:9106/xapi/websocket' --no-check
   ```

   Once the command is executed, make sure you receive the `connection_id` parameter:

   ```
   wscat -c 'wss://<ip_adress>:9106/xapi/websocket' --no-check
   Connected (press CTRL+C to quit)
   < {
     "type":"new_connection",
     "data":{"connection_id":"dc7e0f02-554d-44d4-8579-acaf1e1a8ba7"}
     }
   ```

> **Warning!**
>
> There may be other types of messages `"type"` in the response.

When opening multiple web sockets, each of them is assigned a unique `connection_id`.

2. In order to send an event to Traffic Monitor for analysis, open another terminal tab and send a CURL request. Make sure you specify your data:
   - In line 2, the Traffic Monitor server address
   - In "**-H 'X-API-Auth-Token**" field, replace the sample token with any server token from the `/opt/iw/tm5/etc/config/access_control/tokens.list` file.
   - In "**evt_attributes", "name**": "**capture_ts**" area, specify any past date (for example, this morning). This will be the time of receiving the event in Traffic Monitor.
   - In "**evt_attributes**" "**name**": "**connection_id**" area, specify the value you received in the web socket at step 1.
   - In the last line, specify the path to the file that needs to be attached for analysis.
     Specify the parameters for Swagger authorization:
     - `authToken` (apiKey) – any server token
     - `version` (apiKey) – `1`
     - `companyId` (apiKey) – `IW` (all capitals)

```
curl -k -X 'POST' \
  'https://<ip_adress>:9106/xapi/push/event' \
  -H 'accept: application/json' \
  -H 'X-API-Auth-Token: 7tghcu0jmvwxyg7e7l3l' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-Version: 1' \
  -H 'Content-Type: multipart/form-data' \
  -H 'X-Enable-OpenAPI: 1' \
  -F 'event={
  "evt_class": 4,
  "evt_service": "cloud_storage",
  "evt_attributes": [
    {
      "name": "capture_ts",
      "value": "2025-07-15T07:07:07+03:00"
    },
    {
      "name": "capture_server_ip",
      "value": "127.0.0.1"
    },
    {
      "name": "capture_server_fqdn",
      "value": "localhost"
    },
    {
      "name": "connection_id",
      "value": "4e77e6e5-0ae6-4d07-a660-174e5be42492"
    }
```

```
],
"evt_senders": [
  {
    "identity_id": 1,
    "identity_type": 0,
    "identity_contacts": [
      {
        "name": "auth",
        "value": "user_name@iw"
      }
    ]
  },
  {
    "identity_id": 2,
    "identity_type": 1,
    "identity_contacts": [
      {
        "name": "dnshostname",
        "value": "localhost.computer.domain"
      }
    ]
  }
],
"evt_receivers": [
  {
    "identity_id": 3,
    "identity_type": 3,
    "identity_contacts": []
  }
],
"evt_data": [
  {
    "data_id": 1,
    "data_attributes": [
      {
        "name": "filename",
        "value": "sample.pdf"
      },
      {
        "name": "destination_file_path",
        "value": "goodledrive.com/user/folder/"
      }
    ]
  }
],
"evt_destination": {
  "identity_id": 4,
  "identity_type": 3,
  "identity_contacts": [],
  "identity_attributes": [
    {
      "name": "name",
```

```
        "value": "googledrive.com"
      }
    ]
  }
}' \
  -F 'files[]="/home/work/scripts/sample.pdf";filename=1'
```

3. Once the event has been sent to Traffic Monitor and processed, the web socket will receive a response containing `document_id` and `verdict`:

```
wscat -c 'wss://<ip_adress>:9106/xapi/websocket' --no-check
 Connected (press CTRL+C to quit)
< {
  "type":"new_connection",
  "data":{"connection_id":"dc7e0f02-554d-44d4-8579-acaf1e1a8ba7"}
  }
< {
  "type":"verdict",
  "data":{"document_id":"b509fe5d-bd6a-4647-aea8-0684fde1b369", "verdict":"Allowed"}
  }
```

> **Warning!**
>
> There may be other types of messages `"type"` in the response.

Event verdicts with the corresponding `connection_id` will arrive precisely at the web socket with the specified `connection_id` and only as long as this web socket is open.

11.3 Uploading sample documents and DB unloadings (REST API SDK)

This programming interface is designed to integrate third-party components with Traffic Monitor. Third-party components can generate documents which represent **Database sample unloadings** and **Sample documents** and then pass them to Traffic Monitor using the provided interface. After that, the passed unloadings will be detected in data streams that are analyzed by Traffic Monitor. Rules for detection of unloadings are set in Traffic Monitor console. A third-party component can only upload and update contents of the uploading. The third-party component must be registered in the system according to the procedure that is described in *Registering REST API external component*.

This document will let you realize sample documents and database unloadings automatic download via API. The document is intended for company's developers, technology partners or contractors.



11.3.1 Configuring REST API programming interface

The following parameters participate in configuring the programming interface:

- Address for connection to the SDK service (network address, port);
- Authorization token. A string of characters, which allows you to authorize components that use the SDK. This string can be obtained in the control panel by external automated systems in Traffic Monitor management console;

- Identifier of transmitted data. A string received by a vendor of the external component from InfoWatch. This string describes the type of transmitted data and is used in the licensing policy;
- Identifier of a vendor of the external component. A string received by a vendor of the external component from InfoWatch. This string describes the vendor and is used in the licensing policy.

11.3.2 Registering REST API external component

To integrate an external data interception module with Traffic Monitor system, perform the following steps:

1. Register with InfoWatch company the identifier string of vendor company producing a component. This will link the vendor company with a license that will be installed in the Traffic Monitor system.
2. Get a developer license linked with the company identifier. This will enable loading sample unloadings from this vendor to Traffic Monitor. The developer license allows you to load any sample unloadings.
3. Create an external component registry file as per below. By this moment, the data source names for sample unloading should be already defined, which is to be generated by the external component. The license obtained as per step 2 is added to the registry file.
4. Load the external component registry file to Traffic Monitor system via Traffic Monitor console interface. For details, see the *Administration -> Plugins* section in *InfoWatch Traffic Monitor. User Guide*;
5. Get the access token within the properties of the external component loaded.

Now the System is ready to receive generated sample unloadings from the specified vendor company with the specified source of the sample unloading.

REST API third-party component registry file format

The plugin is an archive in a ZIP format. The archive includes:

- **licenses** folder that contains license files;
- **icon** folder that contains files with pictograms for registered events;
- **manifest.json** file with information on a plugin.

> **Note:**
>
> Folders **licenses** and **icon** are optional. License files and files with pictograms can be located in the root folder.

For events source external systems, **manifest.json** file should contain the following information:

Content	Description
`{`	
`"PLUGIN_ID": "",`	Unique plugin identifier (UUID). Up to 40 characters. Data type: string. Optional. **Example:** `"PLUGIN ID": "189C38D390396EB6E0530100007F1CA200000001",`

Content	Description
`"DISPLAY_NAME": "",`	Plugin name displayed. Data type: string. Optional. **Example:** `"DISPLAY_NAME": "Plugin name",`
`"DESCRIPTION": "",`	Plugin name displayed. Data type: string. **Example:** `"DESCRIPTION": "Test plugin to demonstrate",`
`"VERSION": "",`	Plugin version. Data type: string. Optional. **Example:** `"VERSION": "1.0.0",`
`"VENDOR": "",`	Software company ID in accordance with the company name specified in the license. Data type: string. Optional. **Example:** `"VENDOR": "infowatch",`
`"LICENSE": [ { "PATH":"" } ],`	License files. Data type: array. The paths to license files hould be specified in accordance with archive root folder. Data type: string. Optional. **Example:** `"LICENSE": [ { "PATH":"licenses/license file 1.license" }, { "PATH":"licenses/license file 2.license" }, ],`
`"PATTERN_SEARCH_LICENSE": "",`	Template to search previously downloaded licenses to be binded to plugin. It has the following format: `{operator:or|and|not,child[{name:value}]}"` Data type: string. **Example operator:or**

Content	Description

	<pre>"PATTERN_SEARCH_LICENSE": {
	    "operator": "or",
	    "conditions": [{
	            "origin": "dm"
	        }, {
	            "origin": "dmmobile"
	        }
	    ]
	},</pre>
	Example operator:and
	<pre>"PATTERN_SEARCH_LICENSE": {
	    "operator": "or",
	    "conditions": [
	        {
	            "operator": "and",
	            "conditions": [
	                {"key1": "value1"},
	                {"key2": "value2"},
	                {"key3": "value3"}
	            ]
	        },
	        {
	            "operator": "and",
	            "conditions": [
	                {"key1": "value1"},
	                {"key2": "value2"},
	                {"key3": "value3"}
	            ]
	        }
	    ]
	}</pre>
	Example operator:not
	<pre>"PATTERN_SEARCH_LICENSE": {
	    "operator": "and",
	    "conditions": [
	        {"key1": "value1"},
	        {"key2": "value2"},
	        {"key3": "NONE"},
	        {
	            "operator": "not",
	            "conditions": [
	                {"key4": null}
	            ]
	        }
	    ]
	}</pre>

Content	Description
`"DATATYPE": [ { "VALUE": "" } ]`	List of data types specifying the source systems. Data type: • `"DATATYPE"` - array; • `"VALUE"` - string. Optional. **Example:** `"DATATYPE": [ { "VALUE": "type1_s1", "VALUE": "type2_s2" } ],`
`}`	

Warning!

The file contains an expired license. To use the sample file for registration of a plugin, you need to obtain a new license.

11.3.3 Format of sample unloading and REST API sample document

A sample unloading is a sequence of rows in UTF-8 encoding. The rows should be separated by **CRLF** or **LF** characters. Cell values in the row should be separated by **TAB** or **‘,’** (comma) characters. These limitations follow **TSV** (Tab separated values) **CSV** (Comma separated values) format. The number of cells in all rows must be the same. When you create an unloading, the number of cells must be specified. It is equal to the number of column names, which are passed to the *Create* function. It is only possible to upload contents to an existing sample unloading if the number of cells is equal to the number specified during the unloading creation. If a different number of cells is specified it will make the unloading invalid.

A sample document is a file of any format. If the file is not an encrypted archive, the System extracts the files from the archive, process them with separately from each other and creates a separate sample document basing on each extracted file (a sample document cannot be created based on the whole archive). All created sample documents are placed into the catalog, selected while creating a sample document. If the file is an encrypted archive, sample document cannot be created by System. If the file is an *envelope*, System will create one sample document basing on the file.

11.3.4 Functional description of REST API programming interface

The programming interface is a REST API, that is a sequence of GET/POST requests to a specific resource of a web server and responses to them. The request and response formats are described below. Data protection during the transfer is carried out through the SSL/TLS protocol without the client's authorization. Transferring unencrypted data is not supported. For data transfer, a UTF-8 encoding is used.

Common headings for GET/POST requests

Each request to the server must contain the following obligatory headings:

- X-API-Auth-Token – the authorization token is specified;
- X-API-DataType – an identifier of the transferred data type is specified;
- X-API-CompanyId – an identifier of the vendor is specified;
- X-API-Version – a version of the programming interface is specified. For now, we use number 1.

Common errors

Description of possible errors:

- not_unique_field – not a unique value;
- unique – a unique value;
- too_short – too short value;
- too_long – too long value;
- wrong_length – the value has incorrect length;
- repeat_exactly – the value does not match the value in a different field;
- required – the field value is required;
- not_exist_in_list – the value is not in the list;
- invalid – invalid value;
- contains_child – the node has child elements;
- contains_items – the category node contains entities;
- parent_own_child – the parent contains child elements;
- parent_self_set_current – the object tries to set itself its parent;
- not_valid_json – invalid JSON;
- invalid_cron_format – incorrect cron format;
- read_only – the object is readonly;
- empty – the value cannot be empty.

Successful response

The return code is 200.

Response format:

```
{
    "data": "mixed",

    "meta": "mixed"
}
```

Data/version validation error

The return code is 400.

Error format in case of incorrect version of API (JSON Scheme)

```
{
```

```
    "properties": {
        "error": {
            "description": "Error container",
            "properties":  {
                "code":       {
                    "description": "General description of error",
                    "type":        "string",
                    "value":       "unsupported_version"
                },
                "meta": {
                    "description": "Additional information on error",
                    "type":        "object",
                    "properties":  {
                        "supported_versions": {
                            "description": "Array of supported versions",
                            "type":        "array",
                            "items": {
                                "description": "Supported version",
                                "type": "string"
                            }
                        }
                    }
                }
            }
        }
    }
}
```

Response format in case of request validation error (JSON Scheme)

```
{
```

```
"properties": {

    "error": {

        "description": "Error container",

        "properties":  {

            "code":       {

                "description": "General description of error",

                "type":        "string",

                "value":       "validation"

            },

            "meta": {

                "description": "Additional information on error",

                "type":        "object",

                "properties":  {

                    "validation": {

                        "description": "Names of model fields",

                        "type":        "object",

                        "properties":  {

                            "field":     {

                                "description": "Model field",

                                "type":        "array",

                                "items":       {

                                    "description": "Error",

                                    "type":        "string",

                                    "values": [

                                        "not_unique_field",

                                        "too_short",

                                        "too_long",

                                        "wrong_length",

                                        "repeat_exactly",
```

```
                                        "required",
                                        "not_exist_in_list",
                                        "invalid",
                                        "contains_child",
                                        "contains_items",
                                        "parent_own_child",
                                        "unique",
                                        "not_valid_json",
                                        "invalid_cron_format",
                                        "read_only",
                                        "parent_self_set_current",
                                        "empty"
                                    ]
                                }
                            },
                            "sub_model": {
                                "description": "Name of nested model",
                                "type":        "object",
                                "$ref":       "#/properties/error/properties/meta/properties/
validation"
                            }
                        }
                    }
                }
            }
        }
    }
}
```

Example of error when incorrect version is specified

```
{
    "error": {
        "code": "unsupported_version",
        "meta": {
            "supported_versions": ["2", "3"]
        }
    }
}
```

Example of error when checking uniqueness of a sample unloading name

```
{
    "error": {
        "code": "validation",
        "meta": {
            "validation":    {
                "DISPLAY_NAME": ["unique"]
            },
            "unique_models": [
                {
                    "FINGERPRINT_ID": "0123456789ABCDEF",
                    "DISPLAY_NAME":   "test"
                }
            ]
        }
    }
}
```

Authorization is required

The return code is 401.

Description: is returned in all requests which require authorization.

Response body: no.

Insufficient permissions

The return code is 403.

Description: is returned for all requests if the current user does not have sufficient permissions to perform a given operation.

Response body: no.

Entity is not found

The return code is 404.

Description: is returned if a particular model is not found in the system. For example, a sample unloading is not found.

Response body: no.

Request body is too large

The return code is 413.

Description: is returned if a too large request is transferred.

Response body: no.

Number of server requests is exceeded

The return code is 429.

Description: is returned if too many requests have been sent to the server over a certain period. May include Retry-After heading which specifies in how much time you can repeat the request.

Response body (JSON Scheme):

```
{

    "properties": {

        "error": {

            "description": "Error container",

            "properties":  {

                "code": {

                    "description": "General description of error",

                    "type":        "string",

                    "value":       "too_many_requests "

                },

                "meta": {
```

```
                    "description": "Additional information on error",
                    "type":        "object",
                    "properties":  {
                        "time": {
                            "description": "HTTP date or an integer in seconds, are which you can repeat
the request",
                            "type":        "string"
                        }
                    }
                }
            }
        }
    }
}
```

System error

The return code is 500.

Description: is returned if a logical error has occurred in the system, and the request execution cannot be continued.

Response body (JSON Scheme):

```
{
    "properties": {
        "error": {
            "description": "Error container",
            "properties":  {
                "code": {
                    "description": "General description of error",
                    "type":        "string"
                },
                "meta": {
```

```
                    "description": "Additional information on error",

                    "type":        "object"

                }

            }

        }

    }

}
```

Server is temporarily unavailable

The return code is 503.

Description: informs that currently is not possible to execute requests and the server is down for maintenance. The Retry-After heading contains estimated time of completion of maintenance (the heading value can be either HTTP-date or an integer in seconds).

Response body (JSON Scheme):

```
{

    "properties": {

        "error": {

            "description": "Error container",

            "properties":  {

                "code": {

                    "description": "General description of error",

                    "type":        "string",

                    "value":       "maintains"

                },

                "meta": {

                    "description": "Additional information on error",

                    "type":        "object",

                    "properties":  {

                        "time": {

                            "description": "HTTP-date or integer in seconds after which you can repeat the
request",
```

```
                        "type":        "string"

                    }

                }

            }

        }

    }

}
```

Working with configuration

After the content of the sample unloading has been sent to the SDK and successfully compiled, you can initiate distribution of the modified configuration to the analysis systems.

Distribution of REST API configuration

Description: the method is required to distribute configuration to analysis subsystems.

Method features: POST, synchronous.

Resource: xapi/configuration.

JSON Scheme request body:

```
{

    "STATUS": "distribute"

}
```

In case of success – **the return code** is 200 and **the response body** is as follows:

```
{

    "STATUS": "distributed"

}
```

In case of an error – **the return code** is 500. Possible values of the *code* field:

- error – cannot apply the configuration.

Working with sample unloadings

The chapter contains the following information:

- Creating the unloading;
- Updating data of the unloading;

- Adding data for the unloading;
- Getting state of the unloading;
- Getting current version.

Creating the unloading

Description: the method is required to add a new unloading without filling it with data. The data will be added with a separate method.

Method features: POST, synchronous.

Resource: xapi/etalonTable.

JSON Scheme request body:

```
{

    "properties": {

        "DISPLAY_NAME": {

            "description": "Name of unloading",

            "type": "string",

            "errors": ["required", "not_match", "not_unique_field"]

        },

        "CONDITION_COLUMNS": {

            "description": "a string, an encoded JSON array with column names, e.g. [\"first_column\",
\"second_column\"]",

            "type": "string",

            "errors": ["required", "not_valid_json"]

        },

        "NOTE": {

            "description": "Arbitrary description of the unloading",

            "type": "string"

        }

    },

    "required": [

        "DISPLAY_NAME",

        "CONDITION_COLUMNS"

    ]
```

```
}
```

In case of success – **the return code** is 200. and the response body is as follows:

The response body:

```
{
    "FINGERPRINT_ID":    "string",
    "TYPE":              "string",
    "SOURCE":            "string",
    "DISPLAY_NAME":      "string",
    "NOTE":              "string",
    "CONDITION_COLUMNS": "string",
    "CREATE_DATE":       "string",
    "CHANGE_DATE":       "string",
    "STATUS":            "string"
}
```

Warning!

The number of columns that was specified when creating the unloading cannot be changed. When loading the contents, the number of columns in the rows being loaded should match the value that was specified when creating the sample unloading.

Updating data of the unloading

Description: the method is required to update the unloading contents. All entries of the sample unloading will be deleted, and then, a new entry will be added. This method can be used for the initial filling of the unloading.

Warning!

The filling should have an integer number of rows.
The minimum number of row for filling is 2.
The minimum number of columns is 2.
The minimum byte size of loadable content is 128 byte.
The minimum byte size of loadable content is 2GB.

Methods features: POST, asynchronous.

Resource: `xapi/etalonTable/{id}/content/replace`.

Parameters:

- `id` – fingerprint_id of the object being edited.

Example: `enatalonTable/0123456789ABCDEF/content/replce`

Content-type of the request body should be not `application/json`, but `multipart/form-data` (for more details, see: *https://tools.ietf.org/html/rfc2388*).

Content-Disposition that will include the file should have the `name` field with `CONTENT` value and `filename` a ttribute with the file name and TSV or CSV extension depending on the file format, separated with Tab or commas respectively.

Possible values of errors for the **CONTENT** field:

- not_valid – the file has an invalid signature;
- not_allowed_file_extension – an unauthorized file extension, for now. only 2 formats are supported: TSV and CSV;
- too_long_size – the file size is too large;
- duplicate – this file already exists in the system.

In case of success – **the return code** is 200.

The response body:

```
{

    "FINGERPRINT_ID":    "string",

    "TYPE":              "string",

    "SOURCE":            "string",

    "DISPLAY_NAME":      "string",

    "NOTE":              "string",

    "CONDITION_COLUMNS": "string",

    "CREATE_DATE":       "string",

    "CHANGE_DATE":       "string",

    "STATUS":            "string"

}
```

After getting the 200 return code, you should take the `FINGERPRINT_ID` field and use it for the checking the unloading state, until the state changes to ***ready***.

Important!

The number of cells in loadable rows should match the number of columns that was specified when creating the unloading.
When you delete the unloading through Traffic Monitor console, it is considered existing until you distribute the configuration, i.e. you can update this unloading and add contents to it.
When loading "erroneous" content, i.e. the content which duplicates an already loaded unloading (or its part) or the content whose rows contain an incorrect number of cells, the whole sample object gets an

invalid status. The external subsystem should stop loading this content and go to the next one. Loading the correct content changes the sample object status to the correct one. Incorrect content never reaches the analysis modules.

Adding data for the unloading

Description: the method is used for adding content to the unloading. All content entries will be saved, and a new entry will be added.

Method features: POST.

Resource: `xapi/etalonTable/{id}/content`.

For details, see *Updating data of the unloading*.

Getting state of the unloading

Description: the method is required to get information about the state of a single unloading.

Method features: GET.

Pecypc: `xapi/etalonTable/{id}`.

Parameters:

- `id` – fingerprint_id of the object being edited.

Example: `etalonTable/0123456789ABCDEF`

The response body:

```
{

    "FINGERPRINT_ID":    "string",

    "TYPE":              "string",

    "SOURCE":            "string",

    "DISPLAY_NAME":      "string",

    "NOTE":              "string",

    "CONDITION_COLUMNS": "string",

    "CREATE_DATE":       "string",

    "CHANGE_DATE":       "string",

    "STATUS":            "string"

    "content":           [

        {

            "CONTENT_ID":   "string",

            "CHECKSUM":     "string",
```

```
            "CONTENT_SIZE": "integer",
            "CONTENT_MIME": "string"
        }
    ]
}
```

The `status` field can have the following values:

- sending – sending data to the compiler;
- sent – data has been sent to the compiler, and compilation has been started;
- compiling – the document is being compiled;
- compiled – the document is ready;
- saving – the document is being saved;
- ready – the document has been saved, it is ready for further changes and does not block applying the configuration;
- fatal_error – an error occurred when saving the document, the operation cannot be repeated or continued;
- duplicate – the `content` data is duplicated (loading the content can be continued);
- not_match_delimiter – rows in the unloading do not contain delimiters (COMMA or TAB);
- wrong_column_count – the number of columns in the row does not match the number of columns that was specified when creating the unloading;
- error – a temporary error, you can try the operation again in some time (better in an hour);
- max_words - the maximum number of words is exceeded;
- max_columns - the maximum number of columns is exceeded.

Getting current version

Description: the method is required for getting supported versions. If the return code is 200, you can continue working with the system.
You can use this method to check a license, a version or the authorization. In case of success, the return code is 200. In case of an error, the error code is returned.

Method features: GET.

Resource: `xapi/version`.

The response body:

```
{
    "data":       [
        {
            "NAME":    "string",
            "STATUS": "supported|deprecated|removed"
        }
```

```
    ]

}
```

An example of response if incorrect version is specified:

```
HTTP/1.1 400 Bad Request

Content-Type: application/json

{

    "error": {

        "code": "unsupported_version",

        "meta": {

            "versions": [

                {

                    "NAME":   "1.0.0",

                    "STATUS": "removed"

                },

                {

                    "NAME":   "2.0.0",

                    "STATUS": "deprecated"

                },

                {

                    "NAME":   "3.0.0",

                    "STATUS": "supported"

                }

            ]

        }

    }

}
```

Working with REST API sample documents

This chapter contains the information as follows:

- Creating REST API sample documents catalogue;
- REST API sample documents catalogues review;
- Editing REST API sample documents catalogue;
- Deleting REST API sample documents catalogue;
- Creating REST API sample document;
- REST API sample documents review;
- Updating and replacing content of REST API sample document;
- REST API sample document deletion.

Creating REST API sample documents catalogue

Method: `POST`

Resource: `etalonDocumentCategory`

Request body:

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {},
    "id": "http://infowatch.com/tm/etalonDocumentCategory/edit",
    "type": "object",
    "properties": {
        "PARENT_CATEGORY_ID": {
            "examples": [
                "AC3DF073B60107890A4C704A10577FE700000000"
            ],
            "title": "Parent GUID categories, if null, the folder of automatic sample documents is set",
            "type": "string"
        },
        "DISPLAY_NAME": {
            "title": "Title",
            "type": "string"
        },
        "DIR_PATH (title can be changed)": {
            "title": "Directory full name",
            "type": "string"
        },
        "FP_BIN_VALUE_THRESHOLD": {
            "title": "Binary data quote index",
            "type": "integer"
        },
        "FP_TEXT_VALUE_THRESHOLD": {
            "title": "Text data quote index",
            "type": "integer"
        },
        "NOTE": {
            "title": "Description",
            "type": [
                "null",
                "string"
            ]
        }
    }
}
```

Response body:


```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {},
    "id": "http://infowatch.com/tm/etalonDocumentCategory",
    "type": "object",
    "properties": {
        "CATEGORY_ID": {
            "examples": [
                "AC3DF073B60107890A4C704A10577FE700000000"
            ],
            "title": "GUID categories",
            "type": "string"
        },
        "PARENT_CATEGORY_ID": {
            "examples": [
                "AC3DF073B60107890A4C704A10577FE700000000"
            ],
            "title": "Parent GUID categories",
            "type": "string"
        },
        "DISPLAY_NAME": {
            "title": "Title",
            "type": "string"
        },
        "DIR_PATH (title can be changed)": {
            "title": "Directory full name",
            "type": "string"
        },
        "FP_BIN_VALUE_THRESHOLD": {
            "title": "Binary data quote index",
            "type": "integer"
        },
        "FP_TEXT_VALUE_THRESHOLD": {
            "title": "Text data quote index",
            "type": "integer"
        },
        "NOTE": {
            "title": "Description",
            "type": [
                "null",
                "string"
            ]
        },
        "OWNER": {
            "title": "Token ID",
            "type": "integer"
        },
        "CREATE_DATE": {
            "examples": [
                "2017-07-04 10:45:45.000000"
            ],
            "title": "Date of creation",
            "type": "string"
        },
        "CHANGE_DATE": {
```

```
            "examples": [
                "2017-07-04 10:45:45.000000"
            ],
            "title": "Date of change",
            "type": "string"
        }
    }
}
```

REST API sample documents catalogues review

Method: `GET`

Resource: `etalonDocumentCategory[ /{category_id}]`

Parameters:

- `filter` – filtering list:
  - `create_date[from]` – start of range for filtering by date of creation in UNIX-timestamp format (including the date, can be used saparately from `create_date[to]`)
  - `create_date[to]` – end of range for filtering by date of creation in UNIX-timestamp format (including the date, can be used saparately from `create_date[from]`)
  - `change_date[from]` – start of range for filtering by date of change in UNIX-timestamp format ((including the date, can be used saparately from `change_date[to]`)
  - `change_date[to]` – end of range for filtering by date of change in UNIX-timestamp format ((including the date, can be used saparately from `change_date[from]`)
  - `display_name[]` – catalogue name, * can be used in the end to enable search by `LIKE`, e.g.: `?filter[display_name][]=foo*`
  - `dir_path[]` – full path to catalogue, * can be used in the end to enable search by `LIKE`, e.g.: `?filter[dir_path][]=foo*`
  - `fingerprint_id []` – sample document GUID, e.g.: `?filter[ fingerprint_id ][]= 82EAE496A1686E407B1162E3C3159999F404F559&filter[fingerprint_id][]=AF8EAC3C3C8532C3C780481B7D8C5B8E68D0148F`
- `with[]` – list of addititional entities to be added to the object:
  - `etalonDocumentsCount` – number oa sample documets in the catalogue
- `sort[]` – field used to sort the result. Possible keys are:
  - `create_date` — `desc/asc`, e.g.: `?sort[create_date]=desc`
  - `change_date` — `desc/asc`, e.g. `: ?sort[change_date]=asc`

Response: Massive of objects, sent when created/edited

Examples:

```
GET /xapi/etalonDocumentCategory?filter[create_date][to]=1&filter[display_name][]=foo*&filter[display_name]
[]=bar&with[]=etalonDocumentsCount&sort[create_date]=desc
GET /xapi/etalonDocumentCategory/FAB85F61531BB9E428088EFC81F266FA3C2959E6
```

Editing REST API sample documents catalogue

Method: PUT

Resource: `etalonDocumentCategory/{category_id}`

Request body:


```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {},
    "id": "http://infowatch.com/tm/etalonDocumentCategory/edit",
    "type": "object",
   "properties": {
        "PARENT_CATEGORY_ID": {
            "examples": [
                "AC3DF073B60107890A4C704A10577FE700000000"
            ],
            "title": "Parent GUID categories, if null, the folder of automatic sample documents is set",
            "type": "string"
        },
        "DISPLAY_NAME": {
            "title": "Title",
            "type": "string"
        },
        "DIR_PATH (title can be changed)": {
            "title": "Directory full name",
            "type": "string"
        },
        "FP_BIN_VALUE_THRESHOLD": {
            "title": "Binary data quote index",
            "type": "integer"
        },
        "FP_TEXT_VALUE_THRESHOLD": {
            "title": "Text data quote index",
            "type": "integer"
        },
        "NOTE": {
            "title": "Description",
            "type": [
                "null",
                "string"
            ]
        }
    }
}
```


Response body:


```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {},
    "id": "http://infowatch.com/tm/etalonDocumentCategory",
    "type": "object",
    "properties": {
        "CATEGORY_ID": {
            "examples": [
                "AC3DF073B60107890A4C704A10577FE700000000"
            ],
            "title": "GUID category,
            "type": "string"
        },
        "PARENT_CATEGORY_ID": {
```

```
            "examples": [
                "AC3DF073B60107890A4C704A10577FE700000000"
            ],
            "title": "Parent GUID category",
            "type": "string"
        },
        "DISPLAY_NAME": {
            "title": "Title",
            "type": "string"
        },
        "DIR_PATH (title can be changed)": {
            "title": "Directory full name",
            "type": "string"
        },
        "FP_BIN_VALUE_THRESHOLD": {
            "title": "Binary data quote index",
            "type": "integer"
        },
        "FP_TEXT_VALUE_THRESHOLD": {
            "title": "Text data quote index",
            "type": "integer"
        },
        "NOTE": {
            "title": "Description,
            "type": [
                "null",
                "string"
            ]
        },
        "OWNER": {
            "title": "Token ID",
            "type": "integer"
        },
        "CREATE_DATE": {
            "examples": [
                "2017-07-04 10:45:45.000000"
            ],
            "title": "Date of creation",
            "type": "string"
        },
        "CHANGE_DATE": {
            "examples": [
                "2017-07-04 10:45:45.000000"
            ],
            "title": "Date of change",
            "type": "string"
        }
    }
}
```

Example of editing CURL catalog:

```
curl 'https://example.com/xapi/etalonDocumentCategory/0123456789ABCDEF123400000000' \
  -X PUT \
  -H 'X-API-Auth-Token: 1792jmclf7fer1ikuhby' \
  -H 'X-API-Version: 1' \
  -H 'X-API-CompanyId: IW' \
```

```
  -H 'X-API-DataType: sap' \
  -H 'Content-Type: application/json' \
  --data-binary
'{"PARENT_CATEGORY_ID":"0123456789ABCDEF123400000000","DISPLAY_NAME":"world.txt","DIR_PATH":"c:\\hello\
\world.txt","FP_BIN_VALUE_THRESHOLD":10,"FP_TEXT_VALUE_THRESHOLD":10,"NOTE":null}'
```

Deleting REST API sample documents catalogue

Method: `DELETE`
Resource: `etalonDocumentCategory /{category_id}`

Creating REST API sample document

It is considered that initially the sample document is created by means of `POST /xapi/ etalonDocument` with status *new*. Then sample document's compilation is done, its status should be checked when getting this document (`GET /xapi/ etalonDocument/{id}`) and monitoring it. The sample document's duplicates (STATUS=duplicate) will be deleted, if received.

Method: `POST`
Resource: `etalonDocument`
Request body:

Fields:

Field name	Description
DISPLAY_NAME	Sample document name (if not specified, to be taken from FILE field)
FILE_PATH*	Path to sample document in source, e.g.: • smb://example.com/top_secret/etalon.docx; • \\example\test\text.rtf, ftp://example.com/foo/bar.txt; • oracle://example.com:1521/database/table/row; • https://examlpe.com/same/html/document. This field is used by automatic loader for searching
NOTE	Sample document description.
BIN_VALUE_THRESHOLD	Binary data quote index (by default, to be taken from the first catalog specified).
TEXT_VALUE_THRESHOLD	Text data quote index (by default, to be taken from the first catalog specified).
CATEGORY_ID*	List of GUID categories, separated by comma, e.g.: "E77D893219A24EEAB3D049804FD86751,C743F88121B8449DAF37291474830679,A7AF8EFCF4D54E9393A9B3D750E5ABBB"
FILE*	File itself (The file is sent, not the path to a file. Server cannot connect to your PC and take the file. Thus, the file is sent via request).

Field name	Description
COMPILE_TYPE*	Type of digital fingerprint: • TEXT — to extract text only • ALL — to extract and compile all types of data

* — field is obligatory

Response: JSON-Schema


```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {},
    "id": "http://infowatch.com/tm/etalonDocument",
    "type": "object",
    "properties": {
        "FINGERPRINT_ID": {
            "examples": [
                "139C7026ED48BE13459669024786F7517E5687AE"
            ],
            "title": "sample document identifier",
            "type": "string"
        },
        "DISPLAY_NAME": {
            "title": "Title",
            "type": "string"
        },
        "FILE_PATH": {
            "title": "File full name",
            "type": "string"
        },
        "FILE_SIZE": {
            "title": "File size in bytes",
            "type": "integer"
        },
        "MIME": {
            "title": "File mime-type",
            "type": "string"
        },
        "BIN_VALUE_THRESHOLD": {
            "title": "Binary data quote index",
            "type": "integer"
        },
        "TEXT_VALUE_THRESHOLD": {
            "title": "Text data quote index",
            "type": "integer"
        },
        "NOTE": {
            "title": "Description",
            "type": [
                "null",
                "string"
            ]
        },
        "OWNER": {
```

```
            "title": "Token ID",
            "type": "integer"
        },
        "STATUS": {
            "examples": [
                "new",
                "compiled",
                "compiling",
                "sending",
                "sent",
                "saving",
                "ready",
                "fatal_error",
                "duplicate",
                "error"
            ],
            "title": "Status of processing",
            "type": "string"
        },
        "CREATE_DATE": {
            "examples": [
                "2017-07-04 10:45:45.000000"
            ],
            "title": "Date of creation",
            "type": "string"
        },
        "CHANGE_DATE": {
            "examples": [
                "2017-07-04 10:45:45.000000"
            ],
            "title": "Date of change",
            "type": "string"
        },
        "categories": {
            "type": "array",
            "title": "Массив категорий",
            "items": {
                "type": "object",
                "properties": {
                    "CATEGORY_ID": {
                        "examples": [
                            "AC3DF073B60107890A4C704A10577FE700000000"
                        ],
                        "title": "GUID categories",
                        "type": "string"
                    },
                    "DISPLAY_NAME": {
                        "title": "Title",
                        "type": "string"
                    },
                    "DIR_PATH (title can be changed)": {
                        "title": "Directory full name",
                        "type": "string"
                    },
                    "FP_BIN_VALUE_THRESHOLD": {
                        "title": "Binary data quote index",
                        "type": "integer"
                    },
```

```
                    "FP_TEXT_VALUE_THRESHOLD": {
                        "title": "Text data quote index",
                        "type": "integer"
                    },
                    "NOTE": {
                        "title": "Description",
                        "type": [
                            "null",
                            "string"
                        ]
                    },
                    "OWNER": {
                        "title": "Token ID",
                        "type": "integer"
                    },
                    "CREATE_DATE": {
                        "examples": [
                            "2017-07-04 10:45:45.000000"
                        ],
                        "title": "Date of creation",
                        "type": "string"
                    },
                    "CHANGE_DATE": {
                        "examples": [
                            "2017-07-04 10:45:45.000000"
                        ],
                        "title": "Date of change",
                        "type": "string"
                    }
                }
            }
        }
    }
}
```

Example

```
POST /xapi/etalonDocument HTTP/1.1
Host: tm.example.com
X-API-Auth-Token: brse99dugp1cdbr69axz
X-API-Version: 1
X-API-DataType: cap
X-API-CompanyId: liw
Content-Type: multipart/form-data; boundary=Asrf456BGe4h
Content-Length: (total amount, including subheaders)
(empty line)
(absent preamble)
--Asrf456BGe4h
Content-Disposition: form-data; name="CATEGORY_ID"
(empty line)
2ABECC84F2360B94E0533D003C0A80AE00000000
--Asrf456BGe4h
Content-Disposition: form-data; name="COMPILE_TYPE"
(empty line)
ALL
--Asrf456BGe4h
Content-Disposition: form-data; name="FILE"; filename="foo.doc"
```

```
Content-Type: application/msword
(empty line)
(binary content of document )
```

Example of creating the file by means of CURL

```
curl -i -X POST \
   -H "Content-Type:multipart/form-data" \
   -H "X-API-Auth-Token:brse99dugp1cdbr69axz" \
   -H "X-API-Version:1" \
   -H "X-API-DataType:cap" \
   -H "X-API-CompanyId:liw" \
   -F "CATEGORY_ID=2ABECC84F2360B94E0533D003C0A80AE00000000" \
   -F "FILE=@\"./foo.doc\";filename=\"foo.doc\"" \
   -F "COMPILE_TYPE=ALL" \
 'https://tm.example.com/xapi/etalonDocument'
```

REST API sample documents review

Method: `GET`
Resource: `etalonDocument[/{fingerprint_id}]`
Parameters:

- `filter` – filtering list:
  - `create_date[from]` – start of range for filtering by date of creation in UNIX-timestamp format (including the date, can be used saparately from `create_date[to]`)
  - `create_date[to]` – end of range for filtering by date of creation in UNIX-timestamp format (including the date, can be used saparately from `create_date[from]`)
  - `change_date[from]` – start of range for filtering by date of change in UNIX-timestamp format ((including the date, can be used saparately from `change_date[to]`)
  - `change_date[to]` – end of range for filtering by date of change in UNIX-timestamp format ((including the date, can be used saparately from `change_date[from]`)
  - `display_name[]` – catalog name, * can be used in the end to enable search by `LIKE`, e.g: `?filter[display_name][]=foo*`
  - `file_path[]` – full path to a file, * can be used in the end to enable search by `LIKE`, e.g: `?filter[file_path][]=foo*`
  - `category_id[]` – GUID categories, e.g: `?filter[category_id][]= 82EAE496A1686E407B1162E3C3159999F404F559&filter[category_id][]=AF8EAC3C3C8532C3C780481B7D8C5B8E68D0148F`
  - `category_path[]` – full path to category path, * can be used in the end to enable search by `LIKE`, e.g: `?filter[category_path][]=bar*`
- `with[]` – list of addititional entities to be added to the object:
  - `categories` – list of categories, where sample document is included
- `sort[]` – field used to sort the result. Possible keys are:
  - `create_date` — `desc/asc`, например: `?sort[create_date]=desc`
  - `change_date` — `desc/asc`, например: `?sort[change_date]=asc`

Response: Massive of objects, sent when created/edited

Examples:

```
GET /xapi/etalonDocument?filter[create_date][to]=1&filter[display_name][]=foo*&filter[display_name]
[]=bar&with[]=categories&sort[create_date]=desc
```

```
GET /xapi/etalonDocument/FAB85F61531BB9E428088EFC81F266FA3C2959E6
```

Updating and replacing content of REST API sample document

Method: POST

Resource: `etalonDocument/{fingerprint_id}/update`

Request body:

Fields:

Field name	Description
DISPLAY_NAME	Sample document name (if not specified, to be taken from FILE field)
FILE_PATH	Path to sample document in source, e.g.: • smb://example.com/top_secret/etalon.docx; • \\example\test\text.rtf; • ftp://example.com/foo/bar.txt; • oracle://example.com:1521/database/table/row; • https://examlpe.com/same/html/document.
NOTE	Sample document description
BIN_VALUE_THRESHOLD	Binary data quote index (by default, to be taken from the first catalogue specified)
TEXT_VALUE_THRESHOLD	Text data quote index (by default, to be taken from the first catalogue specified)
CATEGORY_ID	List of GUID categories, separated by comma, e.g: "E77D893219A24EEAB3D049804FD86751,C743F88121B8449DAF37291474830679,A7AF8EFCF4D54E9393A9B3D750E5ABBB"
FILE	File itself (The file is sent, not the path to a file. Server cannot connect to your PC and takes the file. Thus, the file is sent via request)
COMPILE_TYPE	Type of digital fingerprint: • TEXT — to extract text only; • ALL — to extract and compile all types of data

The fields required to be changed should be sent. The rest fields should be put down.

Response:

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
```

```
"definitions": {},
"id": "http://infowatch.com/tm/etalonDocument",
"type": "object",
"properties": {
    "FINGERPRINT_ID": {
        "examples": [
            "139C7026ED48BE13459669024786F7517E5687AE"
        ],
        "title": "Sample document identifier",
        "type": "string"
    },
    "DISPLAY_NAME": {
        "title": "Title",
        "type": "string"
    },
    "FILE_PATH": {
        "title": "File full name",
        "type": "string"
    },
    "FILE_SIZE": {
        "title": "Size in bytes",
        "type": "integer"
    },
    "MIME": {
        "title": "File mime-type",
        "type": "string"
    },
    "BIN_VALUE_THRESHOLD": {
        "title": "Binary data quote index",
        "type": "integer"
    },
    "TEXT_VALUE_THRESHOLD": {
        "title": "Text data quote index",
        "type": "integer"
    },
    "NOTE": {
        "title": "Description",
        "type": [
            "null",
            "string"
        ]
    },
    "OWNER": {
        "title": "Token ID",
        "type": "integer"
    },
    "STATUS": {
        "examples": [
            "new",
            "compiled",
            "compiling",
            "sending",
            "sent",
            "saving",
            "ready",
            "fatal_error",
            "duplicate",
            "error"
```

```
            ],
            "title": "Status of processing",
            "type": "string"
        },
        "CREATE_DATE": {
            "examples": [
                "2017-07-04 10:45:45.000000"
            ],
            "title": "Date of creation",
            "type": "string"
        },
        "CHANGE_DATE": {
            "examples": [
                "2017-07-04 10:45:45.000000"
            ],
            "title": "Date of change",
            "type": "string"
        },
        "categories": {
            "type": "array",
            "title": "Categories array",
            "items": {
                "type": "object",
                "properties": {
                    "CATEGORY_ID": {
                        "examples": [
                            "AC3DF073B60107890A4C704A10577FE700000000"
                        ],
                        "title": "GUID categories",
                        "type": "string"
                    },
                    "DISPLAY_NAME": {
                        "title": "Title",
                        "type": "string"
                    },
                    "DIR_PATH (title can be changed)": {
                        "title": "Directory full name",
                        "type": "string"
                    },
                    "FP_BIN_VALUE_THRESHOLD": {
                        "title": "Binary data quote index",
                        "type": "integer"
                    },
                    "FP_TEXT_VALUE_THRESHOLD": {
                        "title": "Text data quote index",
                        "type": "integer"
                    },
                    "NOTE": {
                        "title": "Description",
                        "type": [
                            "null",
                            "string"
                        ]
                    },
                    "OWNER": {
                        "title": "Token ID",
                        "type": "integer"
                    },
```

```
                    "CREATE_DATE": {
                        "examples": [
                            "2017-07-04 10:45:45.000000"
                        ],
                        "title": "Date of creation",
                        "type": "string"
                    },
                    "CHANGE_DATE": {
                        "examples": [
                            "2017-07-04 10:45:45.000000"
                        ],
                        "title": "Date of change",
                        "type": "string"
                    }
                }
            }
        }
    }
}
```

REST API sample document deletion

Method: DELETE
Resource: `etalonDocument/{fingerprint_id}`

11.4 Access to InfoWatch Traffic Monitor events for external systems (DataExport API SDK)

This application programming interface is intended to integrate Traffic Monitor with external data receiving systems. As a part of DataExport API implementation, external systems get the content and metadata of events, organizational structure data, lists of the Management Console users, and audit data of their actions from Traffic Monitor. DataExport API supports two-way data exchange between Traffic Monitor and external systems.

11.4.1 DataExport API functionality

The functionality of DataExport API includes the following:

- Ability to specify the number of attributes to be included within the result:
    - Event ID;
    - Senders: list of contacts and authentication result, taking into account the potential conflicts;
    - Recipients: list of contacts, and authentication result, taking into account the potential conflicts;
    - Sender's computer, taking into account the potential conflict;
    - Type of event;
    - Date and time of interception;
    - Protocol;
    - Level of violation;
    - Verdict;
    - Path to file or receiver address;
    - Path to file or sender address;
    - User's decision;

        - Path to file;
        - Web resource address;
        - Security objects and catalogues;
        - Event's size;
        - Tags;
        - Set of user attributes logged via plugin;
        - Set of system titles for events;
        - Date and time of event pasting into DB;
        - Name of perimeter where the event has entered;
        - Name of perimeter left by event;
        - List of policies names (including policy ID), triggered by the intercepted object;
        - Group of rules;
        - Interception server;
        - IP interception server;
        - Receiver device name;
        - Sender device name;
        - E-mail subject;
        - Attachment;
        - Mandate level;
        - Mandate category.
- Ability to specify the number of attributes to filtrate:
    - Date and time of interception;
    - Type of event;
    - Protocol;
    - Security objects and catalogues;
    - User's decision;
    - Event's size;
    - Tags;
    - Date and time of event pasting into DB;
    - Group of rules;
    - Level of violation;
    - Verdict;
    - Data and time of the logged user action;
    - Type of the object the action applied to.
- Ability to change the attributes of event in Traffic Monitor:
    - User verdict about an event;
    - User verdict about many events.
- Getting separate event metadata being able to specify the number of attributes to be included into the result. In this regard, if the event contains few enclosed objects, all the extracted text to be assembled into one file;
- Getting all the extracted text of an event;
- Getting a full list of system titles and user attributes of event registered using plugin;
- Getting binary data of a separate event;
- Getting the list of the Management Console users;
- Starting query execution and getting queries' results in Traffic Monitor:
    - list of queries;
    - status of queries;
    - results of queries.
- Retrieving user audit data:
    - Record ID in auditing Console user sessions;

- Data and time of the logged user action in UTC;
- ID of the user who performed the action;
- Full name of the user who performed the action;
- Type of the action the user has taken;
- Type of the object the action applied to;
- ID of the object the action applied to;
- Name of the object the action applied to;
- Description of changes with old and new values.

11.4.2 Authorization to access DataExport API

Authorization to access DataExport API is provided via tokens mechanism, similar to other Traffic Monitor API's. To get token and review DataExport API licensing status it is required adding new type of plugin into the System - Plugin for DataExport API. All the activities related to plugins are performed within the Traffic Monitor console.

Work with plugins

Adding plugin

To add plugin:

1. Go to **Administration - Plugins**;
2. Within the area **Plugins** click .
3. Click **Add**.
4. Review the plugin description and click **Install**.

> **Note:**
>
> To update a plugin, the updated version of this plugin should be added.

Plugin removal

To remove plugin:

1. Go to **Administration - Plugins**;
2. Within the area **Plugins** click 
3. Click **Yes** to confirm the deletion.

Work with tokens

Token is generated automatically after the plugin has been added. All the activities related to tokens are performed within the InfoWatch Traffic Monitor console.

Adding token

To add token:

1. Go to **Administration - Plugins**;

2. Choose the plugin within the **Plugins** area;
3. Go to **Tokens** bookmark;
4. Click .

> **Note:**
>
> Token's value is generated automatically when setting up a token and it cannot be set by user.

Token's data editing

To change token's data:

1. Go to **Administration - Plugins**;
2. Choose the plugin within the **Plugins** area;
3. Go to **Tokens** bookmark;
4. By double left button click, enter the required info within the fields *Name* and *Description*.

Token's value update

If the token was compromised or if there is a potential token leak to the third party, it is required generating a new token's value.

To update token's value:

1. Go to **Administration - Plugins**;
2. Choose the plugin within the **Plugins** area;
3. Go to **Tokens** bookmark;
4. Click .

Copying token

To copy token's value into the clipboard:

1. Go to **Administration - Plugins**;
2. Choose the plugin within the **Plugins** area;
3. Go to **Tokens** bookmark;
4. Click .

Deleting token

To delete token:

1. Go to **Administration - Plugins**;
2. Choose the plugin within the **Plugins** area;
3. Go to **Tokens** bookmark;
4. Click .

11.4.3 DataExport API third-party component registration

To integrate third-party receiving system with InfoWatch Traffic Monitor the following actions are to be taken:

1. Register the authentication type string of company - the component user in Infowatch JSC. Thus letting to connect a user company to the license, to be installed within Traffic Monitor System.
2. Obtain the developer's license related to company's ID. This would let unloading the events data from Traffic Monitor.
3. Create third-party component registry file in accordance with the format below. The license, obtained as per the point 2 is to be added to the file.
4. Download the third-party component registry file to InfoWatch Traffic Monitor System using the Traffic Monitor console interface. Section *Administration - Plugins* (see *InfoWatch Traffic Monitor. User Guide*).
5. Obtain an access token within the downloaded third-party component's properties.

The System is ready to send the company's events data.

DataExport API third-party component registry file format

Plugin is a ZIP format archive. The archive includes:

- **licenses** folder, containing license files;
- **icon** folder, containing the files with pictograms used for registered events;
- **manifest.json** file, containing information on a plugin.

> **Note:**
>
> Folders **licenses** and **icon** are optional. License files and files with pictograms can be located in the root folder.

For external systems, receiving data from Traffic Monitor **manifest.json** file should contain the following information:

Content	Description
`{`	
`"PLUGIN_ID": "",`	Unique plugin identifier (UUID) created by developer. It may include up to 40 characters. The identificator is used for plugin update. Data type: string. Optional. **Example:** `"PLUGIN_ID": "189C38D390396EB6E0530100007F1CA200000001",`
`"DISPLAY_NAME": "",`	Plugin name displayed. Data type: string. Optional.

Content	Description
	Example: `"DISPLAY_NAME": "Plugin name",`
`"DESCRIPTION": "",`	Plugin description displayed. Data type: string. **Example:** `"DESCRIPTION": "Test plugin to demonstrate",`
`"VERSION": "",`	Plugin version. Data type: string. Optional. **Example:** `"VERSION": "1.0.0",`
`"VENDOR": "",`	Software company ID in accordance with the company name specified in the license. Data type: string. Optional. **Example:** `"VENDOR": "infowatch",`
`"LICENSE": [ { "PATH":"" } ],`	License files. Data type: array. The paths to license files should be specified in accordance with archive root folder. Data type: string. Optional. **Example** `"LICENSE": [ { "PATH":"licenses/license file 1.license" }, { "PATH":"licenses/license file 2.license" }, ],`
`"PATTERN_SEARCH_LICENSE": "",`	Template to search previously downloaded licenses to be binded to plugin. It has the following format: `{operator:or|and|not,child[{name:value}]}"` Data type: string. **Example operator:or** `"PATTERN_SEARCH_LICENSE": { "operator": "or",`

Content	Description

	<pre>"conditions": [{
	"origin": "dm"
	}, {
	"origin": "dmmobile"
	}
	]
	}</pre>
	Example operator:and
	<pre>"PATTERN_SEARCH_LICENSE": {
	"operator": "or",
	"conditions": [
	{
	"operator": "and",
	"conditions": [
	{"key1": "value1"},
	{"key2": "value2"},
	{"key3": "value3"}
	]
	},
	{
	"operator": "and",
	"conditions": [
	{"key1": "value1"},
	{"key2": "value2"},
	{"key3": "value3"}
	]
	}
	]
	}</pre>
	Example operator:not
	<pre>"PATTERN_SEARCH_LICENSE": {
	"operator": "and",
	"conditions": [
	{"key1": "value1"},
	{"key2": "value2"},
	{"key3": "NONE"},
	{
	"operator": "not",
	"conditions": [
	{"key4": null}
	]
	}
	]
	}</pre>

`"DATA_RECEIVER": [`	List of identificators for systems receiving data from Traffic Monitor.

Content	Description

`""`
`]`
	Data type:
	• `"DATA_RECEIVER"` - array;
	• `""` - string.
	Optional.
	Example
	`"DATA_RECEIVER": [`
	`"Events reciever1",`
	`"Events reciever2"`
	`]`

`"OBJECT_HEADER": [`	User events attributes. Data type: object.
`{` `"NAME": "",`	Unique attribute's code to be used in the events headers and to be identified in xAPI/PushAPI. If developer's ID is not "IW", the code should start with " `<Software company's ID>_` " prefix. Data type: string. **For example:** CISCO_CALL_DURATION

`"NOTE": {`
`"": "",`
`"": ""`
`},`
	Attribute's name, displayed on Traffic Monitor management console. Data type: object, string.

`"TYPE": "",` `"FORMAT": "",`	Type of value. Data type: enum. Recording format. Possible values: number (integer, fractional), string, date and time UTC + specifying the time zone shifting, duration, hyperlink, listing, logic type. Data type: enum.

`"DATA_CLASS": [""],`
`"USE_IN_POLICY": "",`
`"USE_IN_NOTIFICATION": "",`
	Services, which events will be equipped with new user attribute. Data type: array.
	Using attribute in policies. `"1"` - yes, `"0"` - no. Data type: enum.
	Using attribute in notifications. `"1"` - yes, `"0"` - no. Data type: enum.

`"USE_IN_QUERY": "",`
`"USE_IN_LIST": "",`
`"USE_IN_SHOW": "",`
`"USE_IN_DETAIL": "",`
`"IS_MULTIPLE_VALUE": "",`
`}`
`]`
	Using attribute in queries. `"1"` - yes, `"0"` - no. Data type: enum.
	Displaying attribute in tabular event viewing mode. `"1"` - yes, `"0"` - no. Data type: enum.
	Displaying attribute in brief event viewing form. `"1"` - yes, `"0"` - no. Data type: enum.
	Displaying attribute in detailed event viewing form. `"1"` - yes, `"0"` - no. Data type: enum.
	Multiple value. `"1"` - yes, `"0"` - no. Data type: enum.
	Example

Content	Description

	"OBJECT_HEADER": [
	{
	"NAME": "header_multi_string",
	"NOTE": {
	"rus": "Заголовок строк1",
	"eng": "Header strings1"
	},
	"DATA_CLASS": ["kEmail"],
	"USE_IN_POLICY": "1",
	"USE_IN_QUERY": "1",
	"USE_IN_NOTIFICATION": "1",
	"USE_IN_LIST": "1",
	"USE_IN_SHOW": "1",
	"USE_IN_DETAIL": "1",
	"TYPE": "string",
	"FORMAT": "string",
	"IS_MULTIPLE_VALUE": "1"
	},
	{
	"NAME": "header_date",
	"NOTE": {
	"rus": "Заголовок дата1",
	"eng": "Header date1"
	},
	"DATA_CLASS": ["kEmail"],
	"USE_IN_POLICY": "1",
	"USE_IN_QUERY": "1",
	"USE_IN_NOTIFICATION": "1",
	"USE_IN_LIST": "0",
	"USE_IN_SHOW": "1",
	"USE_IN_DETAIL": "1",
	"TYPE": "string",
	"FORMAT": "date",
	"IS_MULTIPLE_VALUE": "0"
	}
	]

}	

11.4.4 Functional description of DataExport API application programming interface

The chapter contains the following information:

- Common Headers of GET/POST Queries for DataExport API;
- Common errors;
- Getting list of features for the current version of Traffic Monitor;
- Retrieving audit data;
- Getting organizational structure;
- Getting events;
- Retrieving the list of Console users;
- Getting Results of Completed Traffic Monitor Query.

Common Headers of GET/POST Queries for DataExport API

Authorization

Token received for Public API should be specified within **X-API-Auth-Token** header in each API query. The token can be canceled, blocked/deleted via IWTM interface. In this case, the token is considered to be invalid. When the token is invalid, **401** error message will be returned.

Error message example when token is invalid

```
{
    "error": {
        "code": "unauthorized"
    }
}
```

If the token is valid, but API user has no sufficient access rights, **403** error message will be returned.

Error message when user has no sufficient access rights

```
{
    "error": {
        "code": "forbidden"
    }
}
```

Versions

API version is sent via **X-API-Version** header and can be in one of the following states:

- `supported` – the version is supported and actively developing;
- `deprecated` – the version is available but no longer supported. We recommend upgrading to a newer version as this version can be removed in future releases;
- `removed` – the version is completely removed from the product.

In case the server does not support the version, **400** error message is returned.

Response if the version is incorrect

```
GET /hello HTTP/1.1
Host: iwtm.example.com
X-API-Version: 1
```

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
{
    "error": {
        "code": "unsupported_version",
        "meta": {
            "versions": [
                {
                    "NAME":   "1",
                    "STATUS": "removed"
                },
                {
                    "NAME":   "1.1",
                    "STATUS": "deprecated"
                },
                {
                    "NAME":   "1.2",
                    "STATUS": "deprecated"
                },
                {
                    "NAME":   "1.3",
                    "STATUS": "deprecated"
                },
                {
                    "NAME":   "1.4",
                    "STATUS": "deprecated"
                },
                {
                    "NAME":   "1.5",
                    "STATUS": "deprecated"
                },
                {
                    "NAME":   "1.6",
                    "STATUS": "deprecated"
                },
                {
                    "NAME":   "1.7",
                    "STATUS": "deprecated"
                }
                {
                    "NAME":   "1.8",
                    "STATUS": "deprecated"
                }
                {
                    "NAME":   "1.9",
                    "STATUS": "deprecated"
                }
                {
                    "NAME":   "1.10",
                    "STATUS": "supported"
                }
               ]
        }
    }
}
```

Licenses

To operate properly, the module requires a license as follows:

```
{
    "common_name": "iw",
    "importer_name": "dome"
}
```

Where `common_name` – is a name of a licensed company and `importer_name` – is a product, receiving data from TM. **X-API-CompanyId** and **X-API-ProductName** headers should be sent in each query. In case the license is rejected **403** error message will be returned.

Respond when the license is invalid.

```
{
    "error": {
        "code": "licence_not_valid"
    }
}
```

Thus, each server query should contain the following obligatory headings:

- X-API-Version – a version of the programming interface is specified. Current version: **1.10**;
- X-API-CompanyId – an identifier of the vendor is specified, e.g. **iw**;
- X-API-ImporterName – data receiving product is specified, e.g. **dome**;
- X-API-Auth-Token – the authorization token is specified.

Example of query

```
X-API-Version: 1.10
X-API-CompanyId: iw
X-API-ProductName: dome
X-API-Auth-Token: gla20dos1pcdbr69caxz
```

Common errors

Description of possible errors:

- not_unique_field – not a unique value;
- unique – a unique value;
- too_short – too short value;
- too_long – too long value;
- wrong_length – the value has incorrect length;
- repeat_exactly – the value does not match the value in a different field;
- required – the field value is required;
- not_exist_in_list – the value is not in the list;
- invalid – invalid value;
- contains_child – the node has child elements;
- contains_items – the category node contains entities;

- parent_own_child – the parent contains child elements;
- parent_self_set_current – the object tries to set itself its parent;
- not_valid_json – invalid JSON;
- invalid_cron_format – incorrect cron format;
- read_only – the object is readonly;
- empty – the value cannot be empty.

Error	Return code	Описание и формат/тело ответа
Successful response	200	**Response format** `{ "data": "mixed", "meta": "mixed" }`
Data/version validation error	400	**Error format in case of incorrect version of API (JSON Scheme)** `{ "properties": { "error": { "description": "Контейнер ошибки", "properties": { "code": { "description": "Общее описание ошибки", "type": "string", "value": "unsupported_version" }, "meta": { "description": "Дополнительные данные об ошибке", "type": "object", "properties": { "supported_versions": { "description": "Массив поддерживаемых версий", "type": "array", "items": { "description": "Поддерживаемая версия", "type": "string"`

Error	Return code	Описание и формат/тело ответа

		<pre> }
		                    }
		                }
		            }
		        }
		    }
		}</pre>
		Response format in case of request validation error (JSON Scheme)
		<pre>{
		    "properties": {
		        "error": {
		            "description": "Контейнер ошибки",
		            "properties": {
		                "code": {
		                    "description": "Общее
		описание ошибки",
		                    "type": "string",
		                    "value": "validation"
		                },
		                "meta": {
		
		                   "description": "Дополнительные
		данные об ошибке",
		                    "type": "object",
		                    "properties": {
		                        "validation": {
		
		                            "description": "Назван
		ие полей модели",
		
		                            "type": "object
		",
		                             "properties": {
		                                 "field": {
		
		                                        "description":
		 "Поле модели",
		
		                                        "type":
		 "array",
		
		                                        "items":
		{
		
		                                            "descripti
		on": "Ошибка",
		
		                                            "type":
		     "string",</pre>

Error	Return code	Описание и формат/тело ответа
		"values": ["not_unique_field", "too_short", "too_long", "wrong_length", "repeat_exactly", "required", "not_exist_in_list", "invalid", "contains_child", "contains_items", "parent_own_child", "unique", "not_valid_json", "invalid_cron_format", "read_only", "parent_self_set_current", "empty"]

Error	Return code	Описание и формат/тело ответа
		`} }, "sub_model": { "description": "Название вложенной модели", "type": "object", "$ref": "#/properties/error/properties/meta/properties/validation" } } } } } } } }` **Example of error when checking uniqueness of a sample unloading name** `{ "error": { "code": "unsupported_version", "meta": { "supported_versions": ["2", "3"] } } }`
Authorization is required	401	Is returned in all requests which require authorization. **Response body:** no
Insufficient permissions	403	Is returned for all requests if the current user does not have sufficient permissions to perform a given operation. **Response body:** no
Entity is not found	404	Is returned if a particular model is not found in the system. For example, a sample unloading is not found. **Response body:** no
Request body is too large	413	Is returned if a too large request is transferred.

Error	Return code	Описание и формат/тело ответа
		Response body: no

The number of requests to the server has been exceeded	429	Is returned if too many requests have been sent to the server over a certain period. May include Retry-After heading which specifies in how much time you can repeat the request.
		Response body (JSON Scheme)
		{
		"properties": {
		"error": {
		"description": "Контейнер ошибки",
		"properties": {
		"code": {
		"description": "Общее описание ошибки",
		"type": "string",
		"value": "too_many_requests "
		},
		"meta": {
		"description": "Дополнительные данные об ошибки",
		"type": "object",
		"properties": {
		"time": {
		"description": "HTTP-дата, либо целое число в секундах, через которые можно повторить запрос",
		"type": "string"
		}
		}
		}
		}
		}
		}
		}

System error	500	Is returned if a logical error has occurred in the system, and the request execution cannot be continued.
		Response body (JSON Scheme)
		{
		"properties": {
		"error": {

Error	Return code	Описание и формат/тело ответа

		"description": "Контейнер ошибки",
		"properties": {
		"code": {
		"description": "Общее описание ошибки",
		"type": "string"
		},
		"meta": {
		"description": "Дополнительные данные об ошибке",
		"type": "object"
		}
		}
		}
		}
		}

Server is temporarily unavailable	503	Informs that currently is not possible to execute requests and the server is down for maintenance. The Retry-After heading contains estimated time of completion of maintenance (the heading value can be either HTTP-date or an integer in seconds).
		Response body (JSON Scheme)
		{
		"properties": {
		"error": {
		"description": "Контейнер ошибки",
		"properties": {
		"code": {
		"description": "Общее описание ошибки",
		"type": "string",
		"value": "maintains"
		},
		"meta": {
		"description": "Дополнительные данные об ошибке",
		"type": "object",
		"properties": {
		"time": {
		"description": "HTTP-дата, либо цело число в секундах, через которые можно повторить запрос",
		"type": "string"
		}

Error	Return code	Описание и формат/тело ответа

		<pre>}
		}
		}
		}
		}
		}</pre>

Getting list of features for the current version of Traffic Monitor

Description: List of features for the current version of Traffic Monitor.

Method: `GET`

Resource: version/features

Parameters:

Example of request

```
curl -k -i -X GET    -H "X-Api-Auth-Token:1lap4tvpakbgsbj8tyeu"    -H "X-Api-Version:1.3"    -H "X-Enable-Openapi:1"    -H "X-API-CompanyId:iw"    -H "X-API-ProductName:dome"  'https://server.company.com/xapi/version/features'
```

Example of request

```
{
  "data": [
    {
      "VERSION_TM": "7.3.0.285",
      "VERSION_API": "1.6",
      "FEATURES": "sso - single sign-on from vision to tm"
    }
  ]
}
```

Retrieving audit data

> **Note:**
>
> New actions can become available in API for retrieving audit data, while the version of API may remain unchanged.

Description: List of audit data sorted by date and time of logged user actions.

Method: `GET`

Resource: `audit`

Parameters:

- `filter` – filter list:
  - `change_date[>]` – start date and time of the logged user action. Are specified in UNIX timestamp format with microtime dot, for example, `filter[change_date][>]=1591368000.123456`;
  - `change_date[<]` – end date and time of the logged user action. Are specified in UNIX timestamp format with microtime dot, for example, `filter[change_date][<]=1591368000.123456`;
  - `change_date[>=]` – start date and time of the logged user action. Are specified in UNIX timestamp format with microtime dot, for example, `filter[change_date][>=]=1591368000.123456`;
  - `change_date[<=]` – end date and time of the logged user action. Are specified in UNIX timestamp format with microtime dot, for example, `filter[change_date][<=]=1591368000.123456`;
  - `entity_type[]` – the type of the entity to which the action was applied (several values are possible);
  - `operation[]` – operation/user action (several values are possible).
- `without[] -` get data without additional attributes:
  - `property_changes` – data of extended audit.
- `sort[]` – parameter used for sorting:
  - `change_date -` sort by date of `asc` change.
- `start` – a position from which audit data should be delivered (the default value is 0);
- `limit` – how many audit data sets should be delivered (the default value is 100, the value cannot be larger than 1000).

OpenAPI scheme for delivering audit data

```
schemas:
    auditCommon:
      type: object
      required: [AUDIT_LOG_ID, CHANGE_DATE, OPERATION, ENTITY_TYPE]
      properties: &common
        AUDIT_LOG_ID:
          type: integer
          description: Audit log id
        USER_ID:
          type: integer
          nullable: true
          description: |
            User id who performed the action (See /xapi/user).
            USER_ID may be null on login or if changes make by plugin.
        CHANGER_NAME:
          type: string
          nullable: true
          description: |
            User name or Plugin name who performed the action.
            May be null on login.
        CHANGE_DATE:
          type: string
          description: Date and time of the registered user action with format "Y-m-d H:i:s.u" in UTC
          example: "2020-05-14 13:53:52.926560"
        OPERATION:
          type: string
```

```
          description: Operation type over which the action was performed. There may be values not from the
enum.
          enum:
            - run
            - canceled
            - start
            - stop
            - restart
            - view
            - create
            - update
            - change
            - delete
            - delete_ref
            - delete_hash
            - copy
            - move
            - login_failure
            - bruteforce
            - login
            - logout
            - change_password
            - commit
            - rollback
            - draft
            - add
            - import
            - export
            - export_object
            - create_widget
            - update_widget
            - delete_widget
            - download
            - sync
            - decision_update
            - add_tag
            - remove_tag
            - add_role
            - remove_role
            - add_visibility_area
            - remove_visibility_area
            - logging
            - apply_inbound
            - xapi_decision_update
            - start_scan_job
            - apply_result_scan_job
            - update_job_schedule
            - create_log
            - plugin_register
            - plugin_update
            - privileges_update
            - privileges_create
            - notification
            - notification_template
            - notification_test
            - upload_image
            - delete_image
            - create_contact
```

```
            - update_contact
            - delete_contact
            - compile_create
            - compile_update
            - create_template
            - update_template
            - delete_template
            - create_rule
            - update_rule
            - delete_rule
            - notification_test
        ENTITY_TYPE:
          type: string
          description: Entity type over which the action was performed. There may be values not from the
enum.
          enum: *entryTypes
        ENTITY_ID:
          type: string
          nullable: true
          description: |
            ID of the entity which the action was performed. May be GUID or integer as string
            May by null for:
            | OPERATION | ENTITY_TYPE |
            | :--- | :--- |
            | commit | Config |
            | run | QueryReportRun |
            | sync | Adlibitum |
            | login | User |
            | rollback | Config |
          example: "DBAF4F834F7647CBB297BC9DD17B77B100000000"
        ENTITY_DISPLAY_NAME:
          type: string
          nullable: true
          description: The name of the entity over which the action was carried out
          example: "Some name"
        PROPERTY_CHANGES:
          type: object
          nullable: true
          description: Null in default Audit log events
    userLoginLogout:
      type: object
      description: For ENTITY_TYPE User and OPERATION login or logout
      properties:
        <<: *common
        OPERATION:
          type: string
          enum: [login, logout]
        ENTITY_TYPE:
          type: string
          enum: [User]
        PROPERTY_CHANGES:
          type: object
          nullable: true
          required: [request]
          properties:
            request:
              required:
                - hostname
```

```
            - ip
            - result
            - login
          properties:
            hostname:
              type: string
              title: Hostname
              description: Hostname from which authorization passed
            ip:
              type: string
              title: Ip address
              description: Ip address from which authorization passed
            result:
              type: string
              title: Result
              description: Result
              enum:
                - login_failure
                - login
                - logout
            login:
              type: string
              title: User login
              description: Entered user login in input
userBruteForce:
  type: object
  description: For ENTITY_TYPE User and OPERATION login with bruteforce
  properties:
    <<: *common
    OPERATION:
      type: string
      enum: [ bruteforce ]
    ENTITY_TYPE:
      type: string
      enum: [ User ]
    PROPERTY_CHANGES:
      type: object
      nullable: true
      required: [ login ]
      properties:
        request:
          required:
            - login
            - display_name
          properties:
            login:
              type: string
              title: User login
              description: Entered user login in input
            display_name:
              type: string
              title: User full name
              description: User full name
tagPropertyChanges:
  type: object
  properties:
    <<: *common
    ENTITY_TYPE:
```

```
      type: string
      enum: [ Object ]
    OPERATION:
      type: string
      enum: [ add_tag, remove_tag ]
    PROPERTY_CHANGES:
      type: object
      properties:
        new:
          $ref: '#/components/schemas/TagDiffProperties'
        old:
          $ref: '#/components/schemas/TagDiffProperties'
TagDiffProperties:
  type: object
  properties:
    DISPLAY_NAME:
      type: string
    NOTE:
      type: string
userDesicionPropertyChanges:
  type: object
  properties:
    <<: *common
    ENTITY_TYPE:
      type: string
      enum: [ Object ]
    OPERATION:
      type: string
      enum: [ decision_update, xapi_decision_update ]
    PROPERTY_CHANGES:
      type: object
      properties:
        new:
          type: object
          properties:
            USER_DECISION:
              $ref: '#/components/schemas/UserDecisionState'
        old:
          type: object
          properties:
            USER_DECISION:
              $ref: '#/components/schemas/UserDecisionState'
UserDecisionState:
  type: string
  enum: [ NotProcessed, NoViolation, Violation, AdditionalProcessingNeeded ]
objectExportPropertyChanges:
  type: object
  properties:
    <<: *common
    ENTITY_TYPE:
      type: string
      enum: [ Object ]
    OPERATION:
      type: string
      enum: [ export_object ]
    PROPERTY_CHANGES:
      type: object
      properties:
```

```
            new:
              type: object
              properties:
                PARAMS:
                  type: object
                  properties:
                    QUERY_ID:
                      type: integer
                      nullable: true
                    DISPLAY_NAME:
                      type: string
                    COMMENT:
                      type: string
                    FILTER:
                      anyOf:
                        - type: object
                        - type: array
                    SORT:
                      type: object
                    IS_ONE_REPORT:
                      $ref: '#/components/schemas/BooleanAsInteger'
                    ONE_LOAD_REPORT:
                      $ref: '#/components/schemas/BooleanAsInteger'
                    FORMAT:
                      type: string
                      enum: [ simple, full ]
                    TYPE:
                      type: object
                      properties:
                        one_report:
                          type: string
                          enum: [ docx, xls, xlsx, pdf ]
                        several_report:
                          type: string
                          enum: [ docx, pdf ]
                    IS_SEVERAL_REPORT:
                      $ref: '#/components/schemas/BooleanAsInteger'
                    SEVERAL_LOAD_REPORT:
                      $ref: '#/components/schemas/BooleanAsInteger'
                    LOAD_ATTACHMENT:
                      $ref: '#/components/schemas/BooleanAsInteger'
                    KEEP_HIERARCHY:
                      $ref: '#/components/schemas/BooleanAsInteger'
                    LOAD_SNIPPET:
                      $ref: '#/components/schemas/BooleanAsInteger'
                    LOAD_EML:
                      $ref: '#/components/schemas/BooleanAsInteger'
                    LOAD_DEBUG_OBJECT:
                      $ref: '#/components/schemas/BooleanAsInteger'
                    SCOPE:
                      oneOf:
                        - type: string
                          enum: [ all ]
                        - type: array
                          items:
                            type: integer
                    GROUPING:
                      type: array
```

```
                    items:
                      type: string
                      enum: [CaptureDate, Computer, Object, ObjectType, Person]
                  IS_REPORT_ARCHIVE:
                    $ref: '#/components/schemas/BooleanAsInteger'
  BooleanAsStringOrInteger:
    $ref: 'common.yaml#/components/schemas/BooleanAsStringOrInteger'
  BooleanAsInteger:
    $ref: 'common.yaml#/components/schemas/BooleanAsInteger'
  BooleanAsString:
    $ref: 'common.yaml#/components/schemas/BooleanAsString'
```

Example:

Query

```
curl -X GET "https://server.company.com/xapi/audit?sort[change_date]=asc&filter[entity_type]
[]=Adlibitum&filter[entity_type][]=Audit&filter[entity_type][]=Config&filter[entity_type]
[]=Perimeter&filter[entity_type][]=Policy&start=0&limit=5" -H "accept: application/json" -H "X-Enable-
OpenAPI: 1"
```

Response


```
{
  "data": [
    {
      "AUDIT_LOG_ID": 0,
      "USER_ID": 0,
      "CHANGER_NAME": "string",
      "CHANGE_DATE": "2020-05-14 13:53:52.926560",
      "OPERATION": "run",
      "ENTITY_TYPE": "Adlibitum",
      "ENTITY_ID": "DBAF4F834F7647CBB297BC9DD17B77B100000000",
      "ENTITY_DISPLAY_NAME": "Some name",
      "PROPERTY_CHANGES": {}
    },
    {
      "AUDIT_LOG_ID": 0,
      "USER_ID": 0,
      "CHANGER_NAME": "string",
      "CHANGE_DATE": "2020-05-14 13:53:52.926560",
      "OPERATION": "login",
      "ENTITY_TYPE": "User",
      "ENTITY_ID": "DBAF4F834F7647CBB297BC9DD17B77B100000000",
      "ENTITY_DISPLAY_NAME": "Some name",
      "PROPERTY_CHANGES": {
        "request": {
          "hostname": "string",
          "ip": "string",
          "result": "login_failure",
          "login": "string"
        }
      }
    },
    {
```

```
"AUDIT_LOG_ID": 0,
"USER_ID": 0,
"CHANGER_NAME": "string",
"CHANGE_DATE": "2020-05-14 13:53:52.926560",
"OPERATION": "run",
"ENTITY_TYPE": "Policy",
"ENTITY_ID": "DBAF4F834F7647CBB297BC9DD17B77B100000000",
"ENTITY_DISPLAY_NAME": "Some name",
"PROPERTY_CHANGES": {
  "old": {
    "DISPLAY_NAME": "string",
    "NOTE": "string",
    "STATUS": 0,
    "START_DATE": "string",
    "END_DATE": "string",
    "DATA": {
      "ITEMS": [
        {
          "TYPE": "catalog",
          "ID": "string",
          "NAME": "string"
        },
        {
          "TYPE": "fileformat",
          "ID": "string",
          "NAME": "string",
          "MIME_TYPE": "string",
          "ENCRYPTED": 0,
          "MERGED": 0,
          "MAX": 0,
          "MIN": 0,
          "MAX_TYPE": 0,
          "MIN_TYPE": 0
        }
      ]
    }
  },
  "new": {
    "DISPLAY_NAME": "string",
    "NOTE": "string",
    "STATUS": 0,
    "START_DATE": "string",
    "END_DATE": "string",
    "DATA": {
      "ITEMS": [
        {
          "TYPE": "catalog",
          "ID": "string",
          "NAME": "string"
        },
        {
          "TYPE": "fileformat",
          "ID": "string",
          "NAME": "string",
          "MIME_TYPE": "string",
          "ENCRYPTED": 0,
          "MERGED": 0,
          "MAX": 0,
```

```
              "MIN": 0,
              "MAX_TYPE": 0,
              "MIN_TYPE": 0
            }
          ]
        }
      },
      "rule": {
        "ENTITY_TYPE": "PolicyRule",
        "ENTITY_ID": "string",
        "ENTITY_DISPLAY_NAME": "string"
      }
    }
  }
 ],
 "meta": {
   "totalCount": 0,
   "start": 0,
   "limit": 0
 }
}
```


Getting organizational structure

The chapter contains the following information:

- Getting information on groups;
- Getting information on a person;
- Getting information on workstations.

Getting information on groups

The chapter contains the following information:

- Getting the list of groups;
- Getting the list with group contact types;
- Getting statistics on groups.

Getting the list of groups

Description: the list of groups

Method: `GET`

Resource: `group`

Parameters:

- `filter` – the filter list:
  - `change_date[from]` – since the date of the last change in UNIX-timestamp format (can be used separately from `change_date[to]`);
  - `change_date[to]` – up to the date of the last change in UNIX-timestamp format (can be used separately from `change_date[from]`);
  - `create_date[from]` – since the date of the last change in UNIX-timestamp format (can be used separately from `create_date[to]`);
  - `create_date[to]` – up to the date of the last change in UNIX-timestamp format (can be used separately from `create_date[from]`);

    - `is_deleted` – flag 1 or 0. If the parameter is specified, then entities with the specified value will be returned;
    - `disabled` – flag 1 or 0, indicates activity. If the flag is set, then the entity is inactive (the flag is obtained from LDAP);
    - `contact_type[] –` filtering by the contact type (several values can be passed, e.g., `/xapi/group?filter[contact_type][]=webaccount&filter[contact_type][]=icq`);
    - `contact_value[]` – filtering by the contact value (several values can be passed, e.g., `/xapi/group?filter[contact_value][]=user@ad&filter[contact_value][]=login@example.com`);
    - `parent_group_id[]` – filtering by groups (several values can be passed, e.g., `/xapi/group?filter[parent_group_id][]=FCE1409680E5D6469E1F7998266CE9E9&filter[parent_group_id][]=918462B6DC2C954BAAF8706F7DC53F14`).
- `with[]` – the list of additional entities to be added to the object:
    - `contacts` – group contacts;
    - `children` – child groups.
- `sort` – parameter for sorting:
    - `change_date` – sort by the modification date `asc` or `desc` (e.g., `/xapi/group?sort[change_date]=asc`) ⚠️ may slow down returning the result.
- `start` – a start position for returning items, the default value is 0;
- `limit` – how many items to return (the default value is 100, cannot be more than 1000.

Response: The list of persons described with the following *JSON Schema*:

JSON returning schema

```
{
   "$schema" :  "http://json-schema.org/draft-04/schema#" ,
   "type" :  "object" ,
   "required" : [
       "data" ,
       "meta"
   ],
   "properties" : {
       "data" : {
           "description" :  "Array of groups" ,
           "type" :  "array" ,
           "items" : {
               "description" :  "Group" ,
               "type" :  "object" ,
               "required" : [
                   "GROUP_ID" ,
                   "CREATE_DATE" ,
                   "CHANGE_DATE" ,
                   "DISPLAY_NAME" ,
                   "DISTINGUISHEDNAME" ,
                   "GROUP_TYPE" ,
                   "IS_VISIBLE" ,
                   "IS_DELETED" ,
                   "childrenCount"
               ],
```

```
                "properties" : {
                    "GROUP_ID" : {
                        "description" :  "A unique identifier of the group" ,
                        "type" :  "string"
                    },
                    "CREATE_DATE" : {
                        "description" :  "Entry creation date" ,
                        "type" :  "string"
                    },
                    "CHANGE_DATE" : {
                        "description" :  "Entry modification date" ,
                        "type" :  "string"
                    },
                    "DISPLAY_NAME" : {
                        "description" :  "Group name" ,
                        "type" :  "string"
                    },
                    "DISTINGUISHEDNAME" : {
                        "description" :  "Full name" ,
                        "type" :  "string"
                    },
                    "GROUP_TYPE" : {
                        "description" :  "Group type (tmRoot, tmGroup, adRoot, adGroup, adContainer,
adDomain, adOU from LDAP objectclass)" ,
                        "type" :  "string"
                    },
                    "IS_VISIBLE" : {
                        "description" :  "Group visibility" ,
                        "type" :  "integer"
                    },
                    "IS_DELETED" : {
                        "description" :  "Whether the group is deleted from TM" ,
                        "type" :  "integer"
                    },
                    "childrenCount" : {
                        "description" :  "The number of child elements in the group" ,
                        "type" :  "integer"
                    },
                    "contacts" : {
                        "description" :  "Array of contacts" ,
                        "type" :  "array" ,
                        "items" : {
                            "description" :  "Contact" ,
                            "type" :  "object" ,
                            "required" : [
                                "CONTACT_ID" ,
                                "CREATE_DATE" ,
                                "CHANGE_DATE" ,
                                "CONTACT_TYPE" ,
                                "VALUE" ,
                                "IS_PERSONAL" ,
                                "IS_PRIMARY" ,
                                "SOURCE"
                            ],
                            "properties" : {
                                "CONTACT_ID" : {
                                    "description" :  "Object unique identifier" ,
                                    "type" :  "string"
```

```
                                },
                                "CREATE_DATE" : {
                                    "description" :  "Creation date" ,
                                    "type" :  "string"
                                },
                                "CHANGE_DATE" : {
                                    "description" :  "Modification date" ,
                                    "type" :  "string"
                                },
                                "CONTACT_TYPE" : {
                                    "description" :  "Type" ,
                                    "type" :  "string"
                                },
                                "VALUE" : {
                                    "description" :  "Value" ,
                                    "type" :  "string"
                                },
                                "IS_PERSONAL" : {
                                    "description" :  "Belongs to (Work, Personal)" ,
                                    "type" :  "integer"
                                },
                                "IS_PRIMARY" : {
                                    "description" :  "Is primary" ,
                                    "type" :  "integer"
                                },
                                "SOURCE" : {
                                    "description" :  "Type" ,
                                    "type" :  "string"
                                }
                            }
                        }
                    },
                    "children" : {
                        "description" :  "An array of groups which belong to the current group" ,
                        "type" :  "array" ,
                        "items" : {
                            "description" :  "Group" ,
                            "type" :  "object" ,
                            "properties" : {
                                "GROUP_ID" : {
                                    "description" :  "A unique identifier of the group" ,
                                    "type" :  "string"
                                },
                                "PARENT_GROUP_ID" : {
                                    "description" :  "A unique identifier of the parent group" ,
                                    "type" :  "string"
                                },
                                "RELATION_TYPE" : {
                                    "description" :  "Connection type: 0 - tm, 1 - member, 2 -
distinguishedname" ,
                                    "type" :  "integer"
                                }
                            }
                        }
                    }
                }
            },
            "meta" : {
```

```
                "description" :  "Additional data" ,
                "type" :  "object" ,
                "required" : [
                    "totalCount" ,
                    "start" ,
                    "limit"
                ],
                "properties" : {
                    "totalCount" : {
                        "type" :  "integer" ,
                        "description" :  "Total number"
                    },
                    "start" : {
                        "type" :  "integer" ,
                        "description" :  "A position from which returning started"
                    },
                    "limit" : {
                        "type" :  "integer" ,
                        "description" :  "Total number of returned items"
                    }
                }
            }
        }
    }
```

Example:

Query

```
/xapi/group?filter[change_date][from]=1&filter[change_date][to]=2& with []=contacts& with
[]=children&limit=10&sort[change_date]=asc
```

Getting the list with group contact types

Description: The list of all possible types of the group contacts

Method: `GET`

Resource: group/ `contact_type`

- `start` — a start position for returning items, the default value is 0;
- `limit` — how many items to return (the default value is 100, cannot be more than 1000.

Example:

Query

```
GET /xapi/group/contact_type?start=0&limit=10
```

Getting statistics on groups

Description: Statistics on groups

Method: `GET`

Resource: group `/stat`

Parameters:

- `filter` — the filter list:
  - `change_date[from]` — since the date of the last change in UNIX-timestamp format (can be used separately from `change_date[to]`);
  - `change_date[to]` — up to the date of the last change in UNIX-timestamp format (can be used separately from `change_date[from]`).

Response:

JSON returning schema

```
{
     "$schema" :  "http://json-schema.org/draft-04/schema#" ,
     "description" :  "Statistics" ,
     "type" :  "object" ,
     "properties" : {
         "data" : {
             "description" :  "Not used" ,
             "type" :  "array" ,
             "items" : {}
         },
         "meta" : {
             "description" :  "Statistics" ,
             "type" :  "object" ,
             "properties" : {
                 "CHANGE_DATE" : {
                     "description" :  "Modification date" ,
                     "type" :  "object" ,
                     "properties" : {
                         "from" : {
                             "description" :  "The first modification date" ,
                             "type" :  "string"
                         },
                         "to" : {
                             "description" :  "The last modification date" ,
                             "type" :  "string"
                         }
                     }
                 },
                 "totalCount" : {
                     "description" :  "Total number of items" ,
                     "type" :  "integer"
                 }
             }
         }
     }
}
```

Example

```
GET /xapi/group/stat?filter[change_date][to]= 1494951762 &filter[change_date][from]= 1494951000
```

Response

```
{
   "data" : [],
   "meta" : {
     "CHANGE_DATE" : {
       "from" :  "2017-03-24 10:40:03" ,
       "to" :  "2017-05-24 12:10:22"
     },
     "totalCount" : 539693
   }
}
```

Getting information on a person

The chapter contains the following information:

- Getting images of multiple persons;
- Getting person image;
- Getting the list of persons;
- Getting the list of person statuses;
- Getting the list of person contact types;
- Getting statistics on persons.

Getting images of multiple persons

Method: POST

Resource: `person/images`

Parameters (are passed in the form of JSON):

- `response_type` – the response type:
  - `chunked` – `Transfer-Encoding: chunked` is used, where each image is passed in its own block and in the order which is specified in the request body; if there is no image, then null character "\0" is passed instead;
  - `multipart/mixed` – `multipart/mixed` rfc tech-ms, format is used. This format is not part of the HTTP standard, however, it is used in emails, so it can be easily implemented. Besides, it is very similar to `multipart/form-data`. Each image is passed in its block and is separately marked with X-Person-ID title indicating the event the image belongs to. If there is no image, then it is either not passed or the image body is missing.
- `person_id` – an array of person IDs.

Examples:

`chunked` :

Query

```
POST /xapi/person/images
X-API-Version:  1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
(пустая строка)
```

```
{ "response_type" :  "chunked" ,  "person_id" : [ "DEF1CC1FA015CA428D5DE3CD5B804B69" ,
"E277B624BC988C4C96590A8435F634ED" ]}
```

Response

```
HTTP/ 1.1 200 OK
Content-Type: image/png
Transfer-Encoding: chunked
(empty line)
1
\ 0 (no-image character for person DEF1CC1FA015CA428D5DE3CD5B804B69)
48AC
(image content for person E277B624BC988C4C96590A8435F634ED)
0
```

`multipart/mixed` :

Query

```
POST /xapi/person/images
X-API-Version:  1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
(empty line)
{ "response_type" :  "multipart/mixed" ,  "person_id" : [ "DEF1CC1FA015CA428D5DE3CD5B804B69" ,
"E277B624BC988C4C96590A8435F634ED" ]}
```

Response

```
HTTP/ 1.1 200 OK
Content-Type: multipart/mixed; boundary=Asrf456BGe4h
Content-Length: (total volume including titles, can be missing!)
(empty line)
(missing preamble)
--Asrf456BGe4h
Content-Type: application/x-empty
Content-Disposition: attachment; filename= "DEF1CC1FA015CA428D5DE3CD5B804B69"
X-Person-ID: DEF1CC1FA015CA428D5DE3CD5B804B69
(empty line and zero content after)
--Asrf456BGe4h
Content-Disposition: attachment; filename= "E277B624BC988C4C96590A8435F634ED.jpg"
Content-Type: image/jpeg
X-Person-ID: E277B624BC988C4C96590A8435F634ED
(empty line)
(binary contents of the second person E277B624BC988C4C96590A8435F634ED)
--Asrf456BGe4h--
(missing epilogue - empty line)
```

Getting person image

Method: `GET`

Resource: `person/image/{person_id}`

Response: The person image

Getting the list of persons

Description: The list of persons

Method: `GET`

Resource: `person`

Parameters:

- `filter` – the filter list:
  - `change_date[from]` – since the date of the last change in UNIX-timestamp format (can be used separately from `change_date[to]`);
  - `change_date[to]` – up to the date of the last change in UNIX-timestamp format (can be used separately from `change_date[from]`);
  - `create_date[from]` – since the date of the last change in UNIX-timestamp format (can be used separately from `create_date[to]`);
  - `create_date[to]` – up to the date of the last change in UNIX-timestamp format (can be used separately from `create_date[from]`);
  - `is_deleted` – flag 1 or 0. If the parameter is specified, then entities with the specified value will be returned;
  - `disabled` – flag 1 or 0, indicates activity. If the flag is set, then the entity is inactive (the flag is obtained from LDAP);
  - `contact_type[] -` filtering by the contact type (several values can be passed, e.g., `/xapi/person?filter[contact_type][]=webaccount&filter[contact_type][]=icq`);
  - `contact_value[]` – filtering by the contact value (several values can be passed, e.g., `/xapi/person?filter[contact_value ][]=user@ad&filter[contact_value ][]=login@example.com`);
  - `group_id[]` – filtering by groups (several values can be passed, e.g.,: `/xapi/person?filter[group_id][]=webaccount&filter[group_id][]=icq`).
- `with[]` – the list of additional entities to be added to the object:
  - `contacts` – person contact;
  - `p2g` – connection of the person with groups;
  - `status` – person statuses.
- `sort` – parameter for sorting:
  - `change_date` – sort by the modification date `asc` or `desc` (e.g. `/xapi/person?sort[change_date]=asc`) ⚠️ may slow down returning the result.
- `start` – a start position for returning items, the default value is 0;
- `limit` – how many items to return (the default value is 100, cannot be more than 1000.

Response: The list of persons described with the following *JSON Schema*:

JSON schema of returning the list of persons

```
{
    "$schema" :  "http://json-schema.org/draft-04/schema#" ,
    "type" :  "object" ,
    "required" : [
        "data" ,
        "meta"
```

```
    ],
    "properties" : {
        "data" : {
            "description" :  "Array of perons" ,
            "type" :  "array" ,
            "items" : {
                "description" :  "Person" ,
                "type" :  "object" ,
                "required" : [
                    "PERSON_ID" ,
                    "CREATE_DATE" ,
                    "CHANGE_DATE" ,
                    "DISTINGUISHEDNAME" ,
                    "DISPLAY_NAME" ,
                    "GIVENNAME" ,
                    "SN" ,
                    "DISABLED" ,
                    "IS_EMPLOYEE" ,
                    "IS_DELETED"
                ],
                "properties" : {
                    "PERSON_ID" : {
                        "description" :  "Object unique identifier" ,
                        "type" :  "string"
                    },
                    "CREATE_DATE" : {
                        "description" :  "Creation date" ,
                        "type" :  "string"
                    },
                    "CHANGE_DATE" : {
                        "description" :  "Last modification date" ,
                        "type" :  "string"
                    },
                    "DISTINGUISHEDNAME" : {
                        "description" :  "Full name" ,
                        "type" :  "string"
                    },
                    "DISPLAY_NAME" : {
                        "description" :  "First and last names" ,
                        "type" :  "string"
                    },
                    "GIVENNAME" : {
                        "description" :  "First name" ,
                        "type" :  "string"
                    },
                    "SN" : {
                        "description" :  "Last name" ,
                        "type" :  "string"
                    },
                    "DISABLED" : {
                        "description" :  "Person state" ,
                        "type" :  "integer"
                    },
                    "IS_EMPLOYEE" : {
                        "description" :  "If the person is the company employee" ,
                        "type" :  "integer"
                    },
                    "IS_DELETED" : {
```

```
            "description" :  "If the person is deleted from TM" ,
            "type" :  "integer"
        },
        "contacts" : {
            "description" :  "Array of contacts" ,
            "type" :  "array" ,
            "items" : {
                "description" :  "Contact" ,
                "type" :  "object" ,
                "required" : [
                    "CONTACT_ID" ,
                    "CREATE_DATE" ,
                    "CHANGE_DATE" ,
                    "CONTACT_TYPE" ,
                    "VALUE" ,
                    "IS_PERSONAL" ,
                    "IS_PRIMARY" ,
                    "SOURCE"
                ],
                "properties" : {
                    "CONTACT_ID" : {
                        "description" :  "Object unique identifier" ,
                        "type" :  "string"
                    },
                    "CREATE_DATE" : {
                        "description" :  "Creation date" ,
                        "type" :  "string"
                    },
                    "CHANGE_DATE" : {
                        "description" :  "Modification date" ,
                        "type" :  "string"
                    },
                    "CONTACT_TYPE" : {
                        "description" :  "Type" ,
                        "type" :  "string"
                    },
                    "VALUE" : {
                        "description" :  "Value" ,
                        "type" :  "string"
                    },
                    "IS_PERSONAL" : {
                        "description" :  "Belong to (Work, Personal)" ,
                        "type" :  "integer"
                    },
                    "IS_PRIMARY" : {
                        "description" :  "Is primary" ,
                        "type" :  "integer"
                    },
                    "SOURCE" : {
                        "description" :  "Type" ,
                        "type" :  "string"
                    }
                }
            }
        },
        "p2g" : {
            "description" :  "Array of connection with groups" ,
            "type" :  "array" ,
```

```
                        "items" : {
                            "description" :  "Connection with group" ,
                            "type" :  "object" ,
                            "required" : [
                                "PERSON_ID" ,
                                "PARENT_GROUP_ID" ,
                                "RELATION_TYPE"
                            ],
                            "properties" : {
                                "PERSON_ID" : {
                                    "description" :  "Person unique identifier" ,
                                    "type" :  "string"
                                },
                                "PARENT_GROUP_ID" : {
                                    "description" :  "Group unique identifier" ,
                                    "type" :  "string"
                                },
                                "RELATION_TYPE" : {
                                    "description" :  "Connection type: 0 - tm, 1 - member, 2 -
distinguishedname, 3 - primarygroupsid" ,
                                    "type" :  "integer"
                                }
                            }
                        }
                    },
                    "status" : {
                        "description" :  "Array of statuses" ,
                        "type" :  "array" ,
                        "items" : {
                            "description" :  "Person status" ,
                            "type" :  "object" ,
                            "required" : [
                                "IDENTITY_STATUS_ID" ,
                                "CREATE_DATE" ,
                                "CHANGE_DATE" ,
                                "ASSIGNMENT_DATE" ,
                                "DISPLAY_NAME" ,
                                "COLOR" ,
                                "EDITABLE"
                            ],
                            "properties" : {
                                "IDENTITY_STATUS_ID" : {
                                    "description" :  "Status unique identifier" ,
                                    "type" :  "string"
                                },
                                "CREATE_DATE" : {
                                    "description" :  "Status creation date" ,
                                    "type" :  "string"
                                },
                                "CHANGE_DATE" : {
                                    "description" :  "Status modification date" ,
                                    "type" :  "string"
                                },
                                "ASSIGNMENT_DATE" : {
                                    "description" :  "The date when the status was assigned to the
person" ,
                                    "type" :  "string"
                                },
```

```
                                "DISPLAY_NAME" : {
                                    "description" :  "Status name" ,
                                    "type" :  "string"
                                },
                                "COLOR" : {
                                    "description" :  "Status color (e.g. #ff0000)" ,
                                    "type" :  "string"
                                },
                                "EDITABLE" : {
                                    "description" :  "Indicator that the status is not system" ,
                                    "type" :  "integer"
                                }
                            }
                        }
                    }
                }
            }
        },
        "meta" : {
            "description" :  "Additional data" ,
            "type" :  "object" ,
            "required" : [
                "totalCount" ,
                "start" ,
                "limit"
            ],
            "properties" : {
                "totalCount" : {
                    "type" :  "integer" ,
                    "description" :  "Total number"
                },
                "start" : {
                    "type" :  "integer" ,
                    "description" :  "A position from which returning started"
                },
                "limit" : {
                    "type" :  "integer" ,
                    "description" :  "Total number of returned items"
                }
            }
        }
    }
}
```

Example:

Query

```
GET /xapi/person?start= 0 &limit= 10 &filter[change_date][from]= 0 &filter[change_date][to]= 1
&with[]=contacts
```

Response

```
{
    "data" : [{
        "PERSON_ID" :  "532ECB81F75DDC46BE023478C2E86BD9" ,
```

```
"WHENCHANGED" :  "20151229230303.0Z" ,
"WHENCREATED" :  "20151229230303.0Z" ,
"DISABLED" : 0,
"IS_EMPLOYEE" : 1,
"DISTINGUISHEDNAME" :  "CN=Annett Rist,OU=Testou,DC=trafmon,DC=iw" ,
"DISPLAY_NAME" :  "Annett Rist" ,
"GIVENNAME" :  "Annett" ,
"SN" :  "Rist" ,
"OBJECTSID" :  "S-1-5-21-4236050433-3472386798-155597913-156702" ,
"SAMACCOUNTNAME" :  "AC428097" ,
"NETBIOSNAME" :  "TRAFMON" ,
"DNSROOT" :  "trafmon.iw" ,
"USERPRINCIPALNAME" :  "AC428097@acme.com" ,
"MAIL" :  "Annett.Rist@acme.com" ,
"TELEPHONENUMBER" :  "+49-30-626763" ,
"SERVER_ID" :  "65E2D37A643561E5E43C7E757BE121096253B4D6" ,
"CREATE_DATE" :  "2017-03-07 08:54:56.401138" ,
"CHANGE_DATE" :  "2017-03-10 08:40:35.741001" ,
"SOURCE" :  "ad" ,
"HAS_TM_THUMBNAILPHOTO" :  false ,
"HAS_THUMBNAILPHOTO" :  false ,
"TM_PHOTO" :  "da39a3ee5e6b4b0d3255bfef95601890afd80709" ,
"contacts" : [{
    "CONTACT_ID" :  "4A211BA3896A7620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "532ECB81F75DDC46BE023478C2E86BD9" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac428097@acme.com" ,
    "IS_PRIMARY" : 0,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.402060" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.402060" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA389687620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "532ECB81F75DDC46BE023478C2E86BD9" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac428097@trafmon" ,
    "IS_PRIMARY" : 1,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.401926" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.401926" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA389697620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "532ECB81F75DDC46BE023478C2E86BD9" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac428097@trafmon.iw" ,
    "IS_PRIMARY" : 0,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.401998" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.401998" ,
    "SOURCE" :  "ad"
```

```
        }, {
            "CONTACT_ID" :  "4A211BA389667620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "532ECB81F75DDC46BE023478C2E86BD9" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "email" ,
            "VALUE" :  "annett.rist@acme.com" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.401457" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.401457" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA389677620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "532ECB81F75DDC46BE023478C2E86BD9" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "phone" ,
            "VALUE" :  "+49-30-626763" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.401670" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.401670" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3896B7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "532ECB81F75DDC46BE023478C2E86BD9" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "sid" ,
            "VALUE" :  "s-1-5-21-4236050433-3472386798-155597913-156702" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.402130" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.402130" ,
            "SOURCE" :  "ad"
        }],
        "status" : []
    }, {
        "PERSON_ID" :  "92FB7D1EC79A9947938BE582D355B109" ,
        "WHENCHANGED" :  "20151229230304.0Z" ,
        "WHENCREATED" :  "20151229230303.0Z" ,
        "DISABLED" : 0,
        "IS_EMPLOYEE" : 1,
        "DISTINGUISHEDNAME" :  "CN=Lieselene SchÃ¶nemann,OU=Testou,DC=trafmon,DC=iw" ,
        "DISPLAY_NAME" :  "Lieselene SchÃ¶nemann" ,
        "GIVENNAME" :  "Lieselene" ,
        "SN" :  "SchÃ¶nemann" ,
        "OBJECTSID" :  "S-1-5-21-4236050433-3472386798-155597913-156706" ,
        "SAMACCOUNTNAME" :  "AC727058" ,
        "NETBIOSNAME" :  "TRAFMON" ,
        "DNSROOT" :  "trafmon.iw" ,
        "USERPRINCIPALNAME" :  "AC727058@acme.com" ,
        "MAIL" :  "Lieselene.SchÃ¶nemann@acme.com" ,
        "TELEPHONENUMBER" :  "+49-2183-398305" ,
        "SERVER_ID" :  "65E2D37A643561E5E43C7E757BE121096253B4D6" ,
        "CREATE_DATE" :  "2017-03-07 08:54:56.402195" ,
        "CHANGE_DATE" :  "2017-03-10 08:40:35.741198" ,
```

```
"SOURCE" :  "ad" ,
"HAS_TM_THUMBNAILPHOTO" :  false ,
"HAS_THUMBNAILPHOTO" :  false ,
"TM_PHOTO" :  "da39a3ee5e6b4b0d3255bfef95601890afd80709" ,
"contacts" : [{
    "CONTACT_ID" :  "4A211BA389707620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "92FB7D1EC79A9947938BE582D355B109" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac727058@acme.com" ,
    "IS_PRIMARY" : 0,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.403048" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.403048" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA3896E7620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "92FB7D1EC79A9947938BE582D355B109" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac727058@trafmon" ,
    "IS_PRIMARY" : 1,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.402920" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.402920" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA3896F7620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "92FB7D1EC79A9947938BE582D355B109" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac727058@trafmon.iw" ,
    "IS_PRIMARY" : 0,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.402984" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.402984" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA3896C7620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "92FB7D1EC79A9947938BE582D355B109" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "email" ,
    "VALUE" :  "lieselene.schÃ¶nemann@acme.com" ,
    "IS_PRIMARY" : 1,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.402517" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.402517" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA3896D7620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "92FB7D1EC79A9947938BE582D355B109" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "phone" ,
```

```
            "VALUE" :  "+49-2183-398305" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.402731" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.402731" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA389717620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "92FB7D1EC79A9947938BE582D355B109" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "sid" ,
            "VALUE" :  "s-1-5-21-4236050433-3472386798-155597913-156706" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.403117" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.403117" ,
            "SOURCE" :  "ad"
        }],
        "status" : []
    }, {
        "PERSON_ID" :  "F0A3298727D7E0498106AE678CC1F5F8" ,
        "WHENCHANGED" :  "20151229230304.0Z" ,
        "WHENCREATED" :  "20151229230304.0Z" ,
        "DISABLED" : 0,
        "IS_EMPLOYEE" : 1,
        "DISTINGUISHEDNAME" :  "CN=Kasimir Eggers,OU=Testou,DC=trafmon,DC=iw" ,
        "DISPLAY_NAME" :  "Kasimir Eggers" ,
        "GIVENNAME" :  "Kasimir" ,
        "SN" :  "Eggers" ,
        "OBJECTSID" :  "S-1-5-21-4236050433-3472386798-155597913-156710" ,
        "SAMACCOUNTNAME" :  "AC217045" ,
        "NETBIOSNAME" :  "TRAFMON" ,
        "DNSROOT" :  "trafmon.iw" ,
        "USERPRINCIPALNAME" :  "AC217045@acme.com" ,
        "MAIL" :  "Kasimir.Eggers@acme.com" ,
        "TELEPHONENUMBER" :  "+49-7191-389863" ,
        "SERVER_ID" :  "65E2D37A643561E5E43C7E757BE121096253B4D6" ,
        "CREATE_DATE" :  "2017-03-07 08:54:56.403185" ,
        "CHANGE_DATE" :  "2017-03-10 08:40:35.741388" ,
        "SOURCE" :  "ad" ,
        "HAS_TM_THUMBNAILPHOTO" :  false ,
        "HAS_THUMBNAILPHOTO" :  false ,
        "TM_PHOTO" :  "da39a3ee5e6b4b0d3255bfef95601890afd80709" ,
        "contacts" : [{
            "CONTACT_ID" :  "4A211BA389767620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "F0A3298727D7E0498106AE678CC1F5F8" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac217045@acme.com" ,
            "IS_PRIMARY" : 0,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.404114" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.404114" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA389747620E0530100007F388100000000" ,
```

```
        "IDENTITY_ID" :  "F0A3298727D7E0498106AE678CC1F5F8" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "auth" ,
        "VALUE" :  "ac217045@trafmon" ,
        "IS_PRIMARY" : 1,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.403984" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.403984" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389757620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "F0A3298727D7E0498106AE678CC1F5F8" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "auth" ,
        "VALUE" :  "ac217045@trafmon.iw" ,
        "IS_PRIMARY" : 0,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.404055" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.404055" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389727620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "F0A3298727D7E0498106AE678CC1F5F8" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "email" ,
        "VALUE" :  "kasimir.eggers@acme.com" ,
        "IS_PRIMARY" : 1,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.403507" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.403507" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389737620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "F0A3298727D7E0498106AE678CC1F5F8" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "phone" ,
        "VALUE" :  "+49-7191-389863" ,
        "IS_PRIMARY" : 1,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.403722" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.403722" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389777620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "F0A3298727D7E0498106AE678CC1F5F8" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "sid" ,
        "VALUE" :  "s-1-5-21-4236050433-3472386798-155597913-156710" ,
        "IS_PRIMARY" : 1,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.404184" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.404184" ,
        "SOURCE" :  "ad"
```

```
        }],
        "status" : []
    }, {
        "PERSON_ID" :  "52BDB45E49090F4D986899E29FEB35CB" ,
        "WHENCHANGED" :  "20151229230304.0Z" ,
        "WHENCREATED" :  "20151229230304.0Z" ,
        "DISABLED" : 0,
        "IS_EMPLOYEE" : 1,
        "DISTINGUISHEDNAME" :  "CN=Leoni Huck,OU=Testou,DC=trafmon,DC=iw" ,
        "DISPLAY_NAME" :  "Leoni Huck" ,
        "GIVENNAME" :  "Leoni" ,
        "SN" :  "Huck" ,
        "OBJECTSID" :  "S-1-5-21-4236050433-3472386798-155597913-156709" ,
        "SAMACCOUNTNAME" :  "AC716847" ,
        "NETBIOSNAME" :  "TRAFMON" ,
        "DNSROOT" :  "trafmon.iw" ,
        "USERPRINCIPALNAME" :  "AC716847@acme.com" ,
        "MAIL" :  "Leoni.Huck@acme.com" ,
        "TELEPHONENUMBER" :  "+49-9184-802313" ,
        "SERVER_ID" :  "65E2D37A643561E5E43C7E757BE121096253B4D6" ,
        "CREATE_DATE" :  "2017-03-07 08:54:56.404249" ,
        "CHANGE_DATE" :  "2017-03-10 08:40:35.741577" ,
        "SOURCE" :  "ad" ,
        "HAS_TM_THUMBNAILPHOTO" :  false ,
        "HAS_THUMBNAILPHOTO" :  false ,
        "TM_PHOTO" :  "da39a3ee5e6b4b0d3255bfef95601890afd80709" ,
        "contacts" : [{
            "CONTACT_ID" :  "4A211BA3897C7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "52BDB45E49090F4D986899E29FEB35CB" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac716847@acme.com" ,
            "IS_PRIMARY" : 0,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.405104" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.405104" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3897A7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "52BDB45E49090F4D986899E29FEB35CB" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac716847@trafmon" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.404978" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.404978" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3897B7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "52BDB45E49090F4D986899E29FEB35CB" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac716847@trafmon.iw" ,
            "IS_PRIMARY" : 0,
```

```
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.405040" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.405040" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA389787620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "52BDB45E49090F4D986899E29FEB35CB" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "email" ,
            "VALUE" :  "leoni.huck@acme.com" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.404575" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.404575" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA389797620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "52BDB45E49090F4D986899E29FEB35CB" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "phone" ,
            "VALUE" :  "+49-9184-802313" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.404789" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.404789" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3897D7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "52BDB45E49090F4D986899E29FEB35CB" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "sid" ,
            "VALUE" :  "s-1-5-21-4236050433-3472386798-155597913-156709" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.405174" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.405174" ,
            "SOURCE" :  "ad"
        }],
        "status" : []
    }, {
        "PERSON_ID" :  "BE20920FFD837142AAC09C87A874CCD4" ,
        "WHENCHANGED" :  "20151229230304.0Z" ,
        "WHENCREATED" :  "20151229230304.0Z" ,
        "DISABLED" : 0,
        "IS_EMPLOYEE" : 1,
        "DISTINGUISHEDNAME" :  "CN=Adelgund Krebs,OU=Testou,DC=trafmon,DC=iw" ,
        "DISPLAY_NAME" :  "Adelgund Krebs" ,
        "GIVENNAME" :  "Adelgund" ,
        "SN" :  "Krebs" ,
        "OBJECTSID" :  "S-1-5-21-4236050433-3472386798-155597913-156708" ,
        "SAMACCOUNTNAME" :  "AC944381" ,
        "NETBIOSNAME" :  "TRAFMON" ,
        "DNSROOT" :  "trafmon.iw" ,
        "USERPRINCIPALNAME" :  "AC944381@acme.com" ,
        "MAIL" :  "Adelgund.Krebs@acme.com" ,
```

```
"TELEPHONENUMBER" :  "+49-6392-145757" ,
"SERVER_ID" :  "65E2D37A643561E5E43C7E757BE121096253B4D6" ,
"CREATE_DATE" :  "2017-03-07 08:54:56.405238" ,
"CHANGE_DATE" :  "2017-03-10 08:40:35.741765" ,
"SOURCE" :  "ad" ,
"HAS_TM_THUMBNAILPHOTO" :  false ,
"HAS_THUMBNAILPHOTO" :  false ,
"TM_PHOTO" :  "da39a3ee5e6b4b0d3255bfef95601890afd80709" ,
"contacts" : [{
    "CONTACT_ID" :  "4A211BA389827620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "BE20920FFD837142AAC09C87A874CCD4" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac944381@acme.com" ,
    "IS_PRIMARY" : 0,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.406146" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.406146" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA389807620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "BE20920FFD837142AAC09C87A874CCD4" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac944381@trafmon" ,
    "IS_PRIMARY" : 1,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.406026" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.406026" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA389817620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "BE20920FFD837142AAC09C87A874CCD4" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac944381@trafmon.iw" ,
    "IS_PRIMARY" : 0,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.406089" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.406089" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA3897E7620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "BE20920FFD837142AAC09C87A874CCD4" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "email" ,
    "VALUE" :  "adelgund.krebs@acme.com" ,
    "IS_PRIMARY" : 1,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.405584" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.405584" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA3897F7620E0530100007F388100000000" ,
```

```
            "IDENTITY_ID" :  "BE20920FFD837142AAC09C87A874CCD4" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "phone" ,
            "VALUE" :  "+49-6392-145757" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.405836" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.405836" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA389837620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "BE20920FFD837142AAC09C87A874CCD4" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "sid" ,
            "VALUE" :  "s-1-5-21-4236050433-3472386798-155597913-156708" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.406214" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.406214" ,
            "SOURCE" :  "ad"
        }],
        "status" : []
    }, {
        "PERSON_ID" :  "54C067BB3B8518489D098154447242DB" ,
        "WHENCHANGED" :  "20151229230304.0Z" ,
        "WHENCREATED" :  "20151229230304.0Z" ,
        "DISABLED" : 0,
        "IS_EMPLOYEE" : 1,
        "DISTINGUISHEDNAME" :  "CN=Kilian Adolph,OU=Testou,DC=trafmon,DC=iw" ,
        "DISPLAY_NAME" :  "Kilian Adolph" ,
        "GIVENNAME" :  "Kilian" ,
        "SN" :  "Adolph" ,
        "OBJECTSID" :  "S-1-5-21-4236050433-3472386798-155597913-156707" ,
        "SAMACCOUNTNAME" :  "AC598197" ,
        "NETBIOSNAME" :  "TRAFMON" ,
        "DNSROOT" :  "trafmon.iw" ,
        "USERPRINCIPALNAME" :  "AC598197@acme.com" ,
        "MAIL" :  "Kilian.Adolph@acme.com" ,
        "TELEPHONENUMBER" :  "+49-30-399513" ,
        "SERVER_ID" :  "65E2D37A643561E5E43C7E757BE121096253B4D6" ,
        "CREATE_DATE" :  "2017-03-07 08:54:56.406279" ,
        "CHANGE_DATE" :  "2017-03-10 08:40:35.741957" ,
        "SOURCE" :  "ad" ,
        "HAS_TM_THUMBNAILPHOTO" :  false ,
        "HAS_THUMBNAILPHOTO" :  false ,
        "TM_PHOTO" :  "da39a3ee5e6b4b0d3255bfef95601890afd80709" ,
        "contacts" : [{
            "CONTACT_ID" :  "4A211BA389887620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "54C067BB3B8518489D098154447242DB" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac598197@acme.com" ,
            "IS_PRIMARY" : 0,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.407140" ,
```

```
        "CHANGE_DATE" :  "2017-03-07 08:54:56.407140" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389867620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "54C067BB3B8518489D098154447242DB" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "auth" ,
        "VALUE" :  "ac598197@trafmon" ,
        "IS_PRIMARY" : 1,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.407000" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.407000" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389877620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "54C067BB3B8518489D098154447242DB" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "auth" ,
        "VALUE" :  "ac598197@trafmon.iw" ,
        "IS_PRIMARY" : 0,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.407081" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.407081" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389847620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "54C067BB3B8518489D098154447242DB" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "email" ,
        "VALUE" :  "kilian.adolph@acme.com" ,
        "IS_PRIMARY" : 1,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.406603" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.406603" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389857620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "54C067BB3B8518489D098154447242DB" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "phone" ,
        "VALUE" :  "+49-30-399513" ,
        "IS_PRIMARY" : 1,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.406812" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.406812" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389897620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "54C067BB3B8518489D098154447242DB" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "sid" ,
        "VALUE" :  "s-1-5-21-4236050433-3472386798-155597913-156707" ,
        "IS_PRIMARY" : 1,
```

```
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.407217" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.407217" ,
            "SOURCE" :  "ad"
        }],
        "status" : []
    }, {
        "PERSON_ID" :  "D4B1F21E90098D4880ABAB5C81F53F7C" ,
        "WHENCHANGED" :  "20151229230305.0Z" ,
        "WHENCREATED" :  "20151229230305.0Z" ,
        "DISABLED" : 0,
        "IS_EMPLOYEE" : 1,
        "DISTINGUISHEDNAME" :  "CN=Lene Escher,OU=Testou,DC=trafmon,DC=iw" ,
        "DISPLAY_NAME" :  "Lene Escher" ,
        "GIVENNAME" :  "Lene" ,
        "SN" :  "Escher" ,
        "OBJECTSID" :  "S-1-5-21-4236050433-3472386798-155597913-156712" ,
        "SAMACCOUNTNAME" :  "AC673590" ,
        "NETBIOSNAME" :  "TRAFMON" ,
        "DNSROOT" :  "trafmon.iw" ,
        "USERPRINCIPALNAME" :  "AC673590@acme.com" ,
        "MAIL" :  "Lene.Escher@acme.com" ,
        "TELEPHONENUMBER" :  "+49-6131-340459" ,
        "SERVER_ID" :  "65E2D37A643561E5E43C7E757BE121096253B4D6" ,
        "CREATE_DATE" :  "2017-03-07 08:54:56.407282" ,
        "CHANGE_DATE" :  "2017-03-10 08:40:35.742722" ,
        "SOURCE" :  "ad" ,
        "HAS_TM_THUMBNAILPHOTO" :  false ,
        "HAS_THUMBNAILPHOTO" :  false ,
        "TM_PHOTO" :  "da39a3ee5e6b4b0d3255bfef95601890afd80709" ,
        "contacts" : [{
            "CONTACT_ID" :  "4A211BA3898E7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "D4B1F21E90098D4880ABAB5C81F53F7C" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac673590@acme.com" ,
            "IS_PRIMARY" : 0,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.408129" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.408129" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3898C7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "D4B1F21E90098D4880ABAB5C81F53F7C" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac673590@trafmon" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.408006" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.408006" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3898D7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "D4B1F21E90098D4880ABAB5C81F53F7C" ,
            "IDENTITY_TYPE" :  "person" ,
```

```
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac673590@trafmon.iw" ,
            "IS_PRIMARY" : 0,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.408071" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.408071" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3898A7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "D4B1F21E90098D4880ABAB5C81F53F7C" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "email" ,
            "VALUE" :  "lene.escher@acme.com" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.407607" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.407607" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3898B7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "D4B1F21E90098D4880ABAB5C81F53F7C" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "phone" ,
            "VALUE" :  "+49-6131-340459" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.407817" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.407817" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3898F7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "D4B1F21E90098D4880ABAB5C81F53F7C" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "sid" ,
            "VALUE" :  "s-1-5-21-4236050433-3472386798-155597913-156712" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.408198" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.408198" ,
            "SOURCE" :  "ad"
        }],
        "status" : []
    }, {
        "PERSON_ID" :  "7B238377335B7B48AFDC99851D8F4793" ,
        "WHENCHANGED" :  "20151229230305.0Z" ,
        "WHENCREATED" :  "20151229230305.0Z" ,
        "DISABLED" : 0,
        "IS_EMPLOYEE" : 1,
        "DISTINGUISHEDNAME" :  "CN=Annett Schwanke,OU=Testou,DC=trafmon,DC=iw" ,
        "DISPLAY_NAME" :  "Annett Schwanke" ,
        "GIVENNAME" :  "Annett" ,
        "SN" :  "Schwanke" ,
        "OBJECTSID" :  "S-1-5-21-4236050433-3472386798-155597913-156711" ,
        "SAMACCOUNTNAME" :  "AC275412" ,
```

```
"NETBIOSNAME" :  "TRAFMON" ,
"DNSROOT" :  "trafmon.iw" ,
"USERPRINCIPALNAME" :  "AC275412@acme.com" ,
"MAIL" :  "Annett.Schwanke@acme.com" ,
"TELEPHONENUMBER" :  "+49-33633-744751" ,
"SERVER_ID" :  "65E2D37A643561E5E43C7E757BE121096253B4D6" ,
"CREATE_DATE" :  "2017-03-07 08:54:56.408263" ,
"CHANGE_DATE" :  "2017-03-10 08:40:35.742148" ,
"SOURCE" :  "ad" ,
"HAS_TM_THUMBNAILPHOTO" :  false ,
"HAS_THUMBNAILPHOTO" :  false ,
"TM_PHOTO" :  "da39a3ee5e6b4b0d3255bfef95601890afd80709" ,
"contacts" : [{
    "CONTACT_ID" :  "4A211BA389947620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "7B238377335B7B48AFDC99851D8F4793" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac275412@acme.com" ,
    "IS_PRIMARY" : 0,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.409108" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.409108" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA389927620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "7B238377335B7B48AFDC99851D8F4793" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac275412@trafmon" ,
    "IS_PRIMARY" : 1,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.408985" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.408985" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA389937620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "7B238377335B7B48AFDC99851D8F4793" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "auth" ,
    "VALUE" :  "ac275412@trafmon.iw" ,
    "IS_PRIMARY" : 0,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.409051" ,
    "CHANGE_DATE" :  "2017-03-07 08:54:56.409051" ,
    "SOURCE" :  "ad"
}, {
    "CONTACT_ID" :  "4A211BA389907620E0530100007F388100000000" ,
    "IDENTITY_ID" :  "7B238377335B7B48AFDC99851D8F4793" ,
    "IDENTITY_TYPE" :  "person" ,
    "IDENTITY_SOURCE" :  "ad" ,
    "CONTACT_TYPE" :  "email" ,
    "VALUE" :  "annett.schwanke@acme.com" ,
    "IS_PRIMARY" : 1,
    "IS_PERSONAL" : 0,
    "CREATE_DATE" :  "2017-03-07 08:54:56.408589" ,
```

```
            "CHANGE_DATE" :  "2017-03-07 08:54:56.408589" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA389917620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "7B238377335B7B48AFDC99851D8F4793" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "phone" ,
            "VALUE" :  "+49-33633-744751" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.408799" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.408799" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA389957620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "7B238377335B7B48AFDC99851D8F4793" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "sid" ,
            "VALUE" :  "s-1-5-21-4236050433-3472386798-155597913-156711" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.409176" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.409176" ,
            "SOURCE" :  "ad"
        }],
        "status" : []
    }, {
        "PERSON_ID" :  "E277B624BC988C4C96590A8435F634ED" ,
        "WHENCHANGED" :  "20151229230305.0Z" ,
        "WHENCREATED" :  "20151229230305.0Z" ,
        "DISABLED" : 0,
        "IS_EMPLOYEE" : 1,
        "DISTINGUISHEDNAME" :  "CN=Edgar Genz,OU=Testou,DC=trafmon,DC=iw" ,
        "DISPLAY_NAME" :  "Edgar Genz" ,
        "GIVENNAME" :  "Edgar" ,
        "SN" :  "Genz" ,
        "OBJECTSID" :  "S-1-5-21-4236050433-3472386798-155597913-156714" ,
        "SAMACCOUNTNAME" :  "AC938446" ,
        "NETBIOSNAME" :  "TRAFMON" ,
        "DNSROOT" :  "trafmon.iw" ,
        "USERPRINCIPALNAME" :  "AC938446@acme.com" ,
        "MAIL" :  "Edgar.Genz@acme.com" ,
        "TELEPHONENUMBER" :  "+49-7191-731883" ,
        "SERVER_ID" :  "65E2D37A643561E5E43C7E757BE121096253B4D6" ,
        "CREATE_DATE" :  "2017-03-07 08:54:56.409247" ,
        "CHANGE_DATE" :  "2017-03-10 08:40:35.742335" ,
        "SOURCE" :  "ad" ,
        "HAS_TM_THUMBNAILPHOTO" :  false ,
        "HAS_THUMBNAILPHOTO" :  false ,
        "TM_PHOTO" :  "da39a3ee5e6b4b0d3255bfef95601890afd80709" ,
        "contacts" : [{
            "CONTACT_ID" :  "4A211BA3899A7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "E277B624BC988C4C96590A8435F634ED" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
```

```
        "VALUE" :  "ac938446@acme.com" ,
        "IS_PRIMARY" : 0,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.410087" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.410087" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389987620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "E277B624BC988C4C96590A8435F634ED" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "auth" ,
        "VALUE" :  "ac938446@trafmon" ,
        "IS_PRIMARY" : 1,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.409967" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.409967" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389997620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "E277B624BC988C4C96590A8435F634ED" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "auth" ,
        "VALUE" :  "ac938446@trafmon.iw" ,
        "IS_PRIMARY" : 0,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.410030" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.410030" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389967620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "E277B624BC988C4C96590A8435F634ED" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "email" ,
        "VALUE" :  "edgar.genz@acme.com" ,
        "IS_PRIMARY" : 1,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.409572" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.409572" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA389977620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "E277B624BC988C4C96590A8435F634ED" ,
        "IDENTITY_TYPE" :  "person" ,
        "IDENTITY_SOURCE" :  "ad" ,
        "CONTACT_TYPE" :  "phone" ,
        "VALUE" :  "+49-7191-731883" ,
        "IS_PRIMARY" : 1,
        "IS_PERSONAL" : 0,
        "CREATE_DATE" :  "2017-03-07 08:54:56.409781" ,
        "CHANGE_DATE" :  "2017-03-07 08:54:56.409781" ,
        "SOURCE" :  "ad"
    }, {
        "CONTACT_ID" :  "4A211BA3899B7620E0530100007F388100000000" ,
        "IDENTITY_ID" :  "E277B624BC988C4C96590A8435F634ED" ,
        "IDENTITY_TYPE" :  "person" ,
```

```
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "sid" ,
            "VALUE" :  "s-1-5-21-4236050433-3472386798-155597913-156714" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.410155" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.410155" ,
            "SOURCE" :  "ad"
        }],
        "status" : []
    }, {
        "PERSON_ID" :  "DEF1CC1FA015CA428D5DE3CD5B804B69" ,
        "WHENCHANGED" :  "20151229230305.0Z" ,
        "WHENCREATED" :  "20151229230305.0Z" ,
        "DISABLED" : 0,
        "IS_EMPLOYEE" : 1,
        "DISTINGUISHEDNAME" :  "CN=Velten SchÃ¶ning,OU=Testou,DC=trafmon,DC=iw" ,
        "DISPLAY_NAME" :  "Velten SchÃ¶ning" ,
        "GIVENNAME" :  "Velten" ,
        "SN" :  "SchÃ¶ning" ,
        "OBJECTSID" :  "S-1-5-21-4236050433-3472386798-155597913-156713" ,
        "SAMACCOUNTNAME" :  "AC617874" ,
        "NETBIOSNAME" :  "TRAFMON" ,
        "DNSROOT" :  "trafmon.iw" ,
        "USERPRINCIPALNAME" :  "AC617874@acme.com" ,
        "MAIL" :  "Velten.SchÃ¶ning@acme.com" ,
        "TELEPHONENUMBER" :  "+49-9184-565860" ,
        "SERVER_ID" :  "65E2D37A643561E5E43C7E757BE121096253B4D6" ,
        "CREATE_DATE" :  "2017-03-07 08:54:56.410219" ,
        "CHANGE_DATE" :  "2017-03-10 08:40:35.742522" ,
        "SOURCE" :  "ad" ,
        "HAS_TM_THUMBNAILPHOTO" :  false ,
        "HAS_THUMBNAILPHOTO" :  false ,
        "TM_PHOTO" :  "da39a3ee5e6b4b0d3255bfef95601890afd80709" ,
        "contacts" : [{
            "CONTACT_ID" :  "4A211BA389A07620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "DEF1CC1FA015CA428D5DE3CD5B804B69" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac617874@acme.com" ,
            "IS_PRIMARY" : 0,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.411056" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.411056" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3899E7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "DEF1CC1FA015CA428D5DE3CD5B804B69" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac617874@trafmon" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.410937" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.410937" ,
            "SOURCE" :  "ad"
```

```
        }, {
            "CONTACT_ID" :  "4A211BA3899F7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "DEF1CC1FA015CA428D5DE3CD5B804B69" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "auth" ,
            "VALUE" :  "ac617874@trafmon.iw" ,
            "IS_PRIMARY" : 0,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.411000" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.411000" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3899C7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "DEF1CC1FA015CA428D5DE3CD5B804B69" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "email" ,
            "VALUE" :  "velten.schÃ¶ning@acme.com" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.410541" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.410541" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA3899D7620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "DEF1CC1FA015CA428D5DE3CD5B804B69" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "phone" ,
            "VALUE" :  "+49-9184-565860" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.410750" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.410750" ,
            "SOURCE" :  "ad"
        }, {
            "CONTACT_ID" :  "4A211BA389A17620E0530100007F388100000000" ,
            "IDENTITY_ID" :  "DEF1CC1FA015CA428D5DE3CD5B804B69" ,
            "IDENTITY_TYPE" :  "person" ,
            "IDENTITY_SOURCE" :  "ad" ,
            "CONTACT_TYPE" :  "sid" ,
            "VALUE" :  "s-1-5-21-4236050433-3472386798-155597913-156713" ,
            "IS_PRIMARY" : 1,
            "IS_PERSONAL" : 0,
            "CREATE_DATE" :  "2017-03-07 08:54:56.411129" ,
            "CHANGE_DATE" :  "2017-03-07 08:54:56.411129" ,
            "SOURCE" :  "ad"
        }],
        "status" : []
    }],
    "meta" : {
        "totalCount" : 232525,
        "start" : 0,
        "limit" : 10
    }
}
```

Getting the list of person statuses

Description: The list of all possible person statuses

Method: `GET`

Resource: `person/status`

Parameters:

- `start` – a start position for returning items, the default value is 0;
- `limit` – how many items to return (the default value is 100, cannot be more than 1000).

Response: The list of statuses described with the following *JSON Schema*:

JSON schema for returning the list of statuses

```
{
     "$schema" :  "http://json-schema.org/draft-04/schema#" ,
     "type" :  "object" ,
     "properties" : {
         "data" : {
             "description" :  "Array of statuses" ,
             "type" :  "array" ,
             "items" : {
                 "description" :  "Person status" ,
                 "type" :  "object" ,
                 "required" : [
                     "IDENTITY_STATUS_ID" ,
                     "CREATE_DATE" ,
                     "CHANGE_DATE" ,
                     "ASSIGNMENT_DATE" ,
                     "DISPLAY_NAME" ,
                     "COLOR" ,
                     "EDITABLE"
                 ],
                 "properties" : {
                     "IDENTITY_STATUS_ID" : {
                         "description" :  "Status unique identifier" ,
                         "type" :  "string"
                     },
                     "CREATE_DATE" : {
                         "description" :  "Status creation date" ,
                         "type" :  "string"
                     },
                     "CHANGE_DATE" : {
                         "description" :  "Status modification date" ,
                         "type" :  "string"
                     },
                     "ASSIGNMENT_DATE" : {
                         "description" :  "The date when the status was assigned to the person" ,
                         "type" :  "string"
                     },
                     "DISPLAY_NAME" : {
                         "description" :  "Status name" ,
                         "type" :  "string"
                     },
                     "COLOR" : {
```

```
                        "description" :  "Status color (e.g. #ff0000)" ,
                        "type" :  "string"
                    },
                    "EDITABLE" : {
                        "description" :  "Indicator that the status is not system" ,
                        "type" :  "integer"
                    }
                }
            }
        },
        "meta" : {
            "description" :  "Additional data" ,
            "type" :  "object" ,
            "required" : [
                "totalCount" ,
                "start" ,
                "limit"
            ],
            "properties" : {
                "totalCount" : {
                    "description" :  "Total number" ,
                    "type" :  "integer"
                },
                "start" : {
                    "description" :  "A position from which returning started" ,
                    "type" :  "integer"
                },
                "limit" : {
                    "description" :  "Total number of returned items" ,
                    "type" :  "integer"
                }
            }
        }
    }
}
```

Example:

Query
`GET /xapi/person/status?start= 0 &limit= 10`

Getting the list of person contact types

Description: The list of all possible contact types

Method: `GET`

Resource: `person/contact_type`

Parameters:

- `start` – a start position for returning items, the default value is 0;
- `limit` – how many items to return (the default value is 100, cannot be more than 1000).

Response: The list of types described with the following *JSON Schema*:

JSON schema of returning the list of contacts

```
{
     "$schema" :  "http://json-schema.org/draft-04/schema#" ,
     "type" :  "object" ,
     "properties" : {
         "data" : {
             "description" :  "Array of contact types" ,
             "type" :  "array" ,
             "items" : {
                 "description" :  "Contact type" ,
                 "type" :  "object" ,
                 "properties" : {
                     "CONTACT_TYPE_ID" : {
                         "description" :  "A unique identifier of the contact type" ,
                         "type" :  "string"
                     },
                     "DISPLAY_NAME" : {
                         "description" :  "Contact type name" ,
                         "type" :  "string"
                     },
                     "LANGUAGE" : {
                         "description" :  "Translation language" ,
                         "type" :  "string"
                     },
                     "MNEMO" : {
                         "description" :  "Mnemonic identifier (is used in contact entity)" ,
                         "type" :  "string"
                     }
                 }
             }
         },
         "meta" : {
             "description" :  "Additional data" ,
             "type" :  "object" ,
             "required" : [
                 "totalCount" ,
                 "start" ,
                 "limit"
             ],
             "properties" : {
                 "totalCount" : {
                     "description" :  "Total number" ,
                     "type" :  "integer"
                 },
                 "start" : {
                     "description" :  "A position where to start returning items" ,
                     "type" :  "integer"
                 },
                 "limit" : {
                     "description" :  "Total number of returned items" ,
                     "type" :  "integer"
                 }
             }
         }
     }
}
```

Example:

Query

```
GET /xapi/person/contact_type?start= 0 &limit= 10
```

Getting statistics on persons

Description: Statistics on persons

Method: `GET`

Resource: `person/stat`

Parameters:

- `filter` – the filter list:
  - `change_date[from]` – since the date of the last change in UNIX-timestamp format (can be used separately from `change_date[to]`);
  - `change_date[to]` – up to the date of the last change in UNIX-timestamp format (can be used separately from `change_date[from]`).

Response:

JSON returning schema

```
{
    "$schema" :  "http://json-schema.org/draft-04/schema#" ,
    "description" :  "Statistics" ,
    "type" :  "object" ,
    "properties" : {
        "data" : {
            "description" :  "Not used" ,
            "type" :  "array" ,
            "items" : {}
        },
        "meta" : {
            "description" :  "Statistics" ,
            "type" :  "object" ,
            "properties" : {
                "CHANGE_DATE" : {
                    "description" :  "Modification date" ,
                    "type" :  "object" ,
                    "properties" : {
                        "from" : {
                            "description" :  "The first modification date" ,
                            "type" :  "string"
                        },
                        "to" : {
                            "description" :  "The last modification date" ,
                            "type" :  "string"
                        }
                    }
                },
                "totalCount" : {
                    "description" :  "Total number of items" ,
                    "type" :  "integer"
```

```
                }
            }
        }
    }
}
```

Example

```
GET /xapi/person/stat?filter[change_date][to]= 1494951762 &filter[change_date][from]= 1494951000
```

Response

```
{
   "data" : [],
   "meta" : {
     "CHANGE_DATE" : {
       "from" :  "2017-03-24 10:40:03" ,
       "to" :  "2017-05-24 12:10:22"
     },
     "totalCount" : 539693
   }
}
```

Getting information on workstations

The chapter contains the following information:

- Getting the list of workstations;
- Getting the list of workstation statuses;
- Getting the list of workstation contact types;
- Getting statistics on workstations.

Getting the list of workstations

Description: The list of workstations

Method: `GET`

Resource: `workstation`

Параметры:

- `filter` – the filter list:
  - `change_date[from]` – since the date of the last change in UNIX-timestamp format (can be used separately from `change_date[to]`);
  - `change_date[to]` – up to the date of the last change in UNIX-timestamp format (can be used separately from `change_date[from]`);
  - `create_date[from]` – since the date of the last change in UNIX-timestamp format (can be used separately from `create_date[to]`);
  - `create_date[to]` – up to the date of the last change in UNIX-timestamp format (can be used separately from `create_date[from]`);
  - is_deleted – flag 1 or 0. If the parameter is specified, then entities with the specified value will be returned;

- `disabled` – flag 1 or 0, indicates activity. If the flag is set, then the entity is inactive (the flag is obtained from LDAP);
- `contact_type[] –` filtering by the contact type (several values can be passed, e.g., `/xapi/workstation?filter[contact_type][]=webaccount&filter[contact_type][]=icq`);
- `contact_value[]` – filtering by the contact value (several values can be passed, e.g., `/xapi/workstation?filter[contact_value][]=user@ad&filter[contact_value][]=login@example.com`);
- `group_id[]` – filtering by groups (several values can be passed, e.g., `/xapi/workstation?filter[group_id][]=FCE1409680E5D6469E1F7998266CE9E9&filter[group_id][]=918462B6DC2C954BAAF8706F7DC53F14`).

- `with[]` – the list of additional entities to be added to the object:
  - `contacts` – workstation contacts;
  - `w2g` – workstation connections with groups;
  - `status` – workstation statuses.
- `sort` – parameter for sorting:
  - `change_date` – sort by the modification date `asc` or `desc` (e.g., `/xapi/workstation?sort[change_date]=asc`) may slow down returning the result.
- `start` – a start position for returning items, the default value is 0;
- `limit` – how many items to return (the default value is 100, cannot be more than 1000.

Response: The list of workstations described with the following *JSON Schema*:

JSON returning schema

```
{
    "$schema" :  "http://json-schema.org/draft-04/schema#" ,
    "type" :  "object" ,
    "required" : [
        "data" ,
        "meta"
    ],
    "properties" : {
        "data" : {
            "description" :  "Array of workstations" ,
            "type" :  "array" ,
            "items" : {
                "description" :  "Workstation" ,
                "type" :  "object" ,
                "required" : [
                    "WORKSTATION_ID" ,
                    "CREATE_DATE" ,
                    "CHANGE_DATE" ,
                    "DISPLAY_NAME" ,
                    "DISTINGUISHEDNAME" ,
                    "DNSHOSTNAME" ,
                    "DISABLED" ,
                    "TYPE" ,
                    "IS_DELETED"
                ],
                "properties" : {
                    "WORKSTATION_ID" : {
```

```
                        "description" :  "Workstation unique identifier" ,
                        "type" :  "string"
                    },
                    "CREATE_DATE" : {
                        "description" :  "Entry creation date" ,
                        "type" :  "string"
                    },
                    "CHANGE_DATE" : {
                        "description" :  "Entry modification date" ,
                        "type" :  "string"
                    },
                    "DISPLAY_NAME" : {
                        "description" :  "Workstation name" ,
                        "type" :  "string"
                    },
                    "DISTINGUISHEDNAME" : {
                        "description" :  "Distinguished name" ,
                        "type" :  "string"
                    },
                    "DNSHOSTNAME" : {
                        "description" :  "Full name" ,
                        "type" :  "string"
                    },
                    "DISABLED" : {
                        "description" :  "Workstation state" ,
                        "type" :  "integer"
                    },
                    "TYPE" : {
                        "description" :  "Type: computer — Workstation, mobile — Mobile device, terminal —
Terminal server" ,
                        "type" :  "string"
                    },
                    "IS_DELETED" : {
                        "description" :  "If the workstation is deleted from TM" ,
                        "type" :  "integer"
                    },
                    "contacts" : {
                        "description" :  "Array of contacts" ,
                        "type" :  "array" ,
                        "items" : {
                            "description" :  "Contact" ,
                            "type" :  "object" ,
                            "required" : [
                                "CONTACT_ID" ,
                                "CREATE_DATE" ,
                                "CHANGE_DATE" ,
                                "CONTACT_TYPE" ,
                                "VALUE" ,
                                "IS_PERSONAL" ,
                                "IS_PRIMARY" ,
                                "SOURCE"
                            ],
                            "properties" : {
                                "CONTACT_ID" : {
                                    "description" :  "Object unique identifier" ,
                                    "type" :  "string"
                                },
                                "CREATE_DATE" : {
```

```
                                    "description" :  "Creation date" ,
                                    "type" :  "string"
                                },
                                "CHANGE_DATE" : {
                                    "description" :  "Modification date" ,
                                    "type" :  "string"
                                },
                                "CONTACT_TYPE" : {
                                    "description" :  "Type" ,
                                    "type" :  "string"
                                },
                                "VALUE" : {
                                    "description" :  "Value" ,
                                    "type" :  "string"
                                },
                                "IS_PERSONAL" : {
                                    "description" :  "Belongs to (Work, Personal)" ,
                                    "type" :  "integer"
                                },
                                "IS_PRIMARY" : {
                                    "description" :  "Is primary" ,
                                    "type" :  "integer"
                                },
                                "SOURCE" : {
                                    "description" :  "Type" ,
                                    "type" :  "string"
                                }
                            }
                        }
                    },
                    "w2g" : {
                        "description" :  "Array of connections with groups" ,
                        "type" :  "array" ,
                        "items" : {
                            "description" :  "Connection with group" ,
                            "type" :  "object" ,
                            "required" : [
                                "WORKSTATION_ID" ,
                                "PARENT_GROUP_ID" ,
                                "RELATION_TYPE"
                            ],
                            "properties" : {
                                "WORKSTATION_ID" : {
                                    "description" :  "Workstation unique identifier" ,
                                    "type" :  "string"
                                },
                                "PARENT_GROUP_ID" : {
                                    "description" :  "Group unique identifier" ,
                                    "type" :  "string"
                                },
                                "RELATION_TYPE" : {
                                    "description" :  "Connection type: 0 - tm, 1 - member, 2 -
distinguishedname, 3 - primarygroupsid" ,
                                    "type" :  "integer"
                                }
                            }
                        }
                    },
```

```
                    "status" : {
                        "description" :  "Array of statuses" ,
                        "type" :  "array" ,
                        "items" : {
                            "description" :  "Person status" ,
                            "type" :  "object" ,
                            "required" : [
                                "IDENTITY_STATUS_ID" ,
                                "CREATE_DATE" ,
                                "CHANGE_DATE" ,
                                "ASSIGNMENT_DATE" ,
                                "DISPLAY_NAME" ,
                                "COLOR" ,
                                "EDITABLE"
                            ],
                            "properties" : {
                                "IDENTITY_STATUS_ID" : {
                                    "description" :  "Status unique identifier" ,
                                    "type" :  "string"
                                },
                                "CREATE_DATE" : {
                                    "description" :  "Status creation date" ,
                                    "type" :  "string"
                                },
                                "CHANGE_DATE" : {
                                    "description" :  "Status modification date" ,
                                    "type" :  "string"
                                },
                                "ASSIGNMENT_DATE" : {
                                    "description" :  "The date when the status was assigned to the
person" ,
                                    "type" :  "string"
                                },
                                "DISPLAY_NAME" : {
                                    "description" :  "Status name" ,
                                    "type" :  "string"
                                },
                                "COLOR" : {
                                    "description" :  "Status color (e.g. #ff0000)" ,
                                    "type" :  "string"
                                },
                                "EDITABLE" : {
                                    "description" :  "Indicator that the status is not system" ,
                                    "type" :  "integer"
                                }
                            }
                        }
                    }
                }
            }
        },
        "meta" : {
            "description" :  "Additional data" ,
            "type" :  "object" ,
            "required" : [
                "totalCount" ,
                "start" ,
                "limit"
```

```
            ],
            "properties" : {
                "totalCount" : {
                    "type" :  "integer" ,
                    "description" :  "Total number"
                },
                "start" : {
                    "type" :  "integer" ,
                    "description" :  "A position from which returning started"
                },
                "limit" : {
                    "type" :  "integer" ,
                    "description" :  "Total number of returned items"
                }
            }
        }
    }
}
```

Example:

Query

```
/xapi/workstation?filter[change_date][from]=1&filter[change_date][to]=2& with []=contacts& with
[]=w2g&limit=10&sort[change_date]=asc
```

Getting the list of workstation statuses

Description: The list of all possible workstation statuses

Method: `GET`

Resource: `workstation/status`

- `start` – a start position for returning items, the default value is 0;
- `limit` – how many items to return (the default value is 100, cannot be more than 1000).

Example:

Query

```
GET /xapi/workstation/status?start=0&limit=10
```

Getting the list of workstation contact types

Description: The list of all possible types of workstation contacts

Method: `GET`

Resource: `workstation/contact_type`

- `start` – a start position for returning items, the default value is 0;
- `limit` – how many items to return (the default value is 100, cannot be more than 1000).

Example:

Query

```
GET /xapi/workstation/contact_type?start=0&limit=10
```

Getting statistics on workstations

Description: Statistics on workstations

Method: `GET`

Resource: workstation `/stat`

Parameters:

- `filter` – the filter list:
  - `change_date[from]` – since the date of the last change in UNIX-timestamp format (can be used separately from `change_date[to]`);
  - `change_date[to]` – up to the date of the last change in UNIX-timestamp format (can be used separately from `change_date[from]`).

Response:

JSON returning schema

```
{
    "$schema" :  "http://json-schema.org/draft-04/schema#" ,
    "description" :  "Statistics" ,
    "type" :  "object" ,
    "properties" : {
        "data" : {
            "description" :  "Not used" ,
            "type" :  "array" ,
            "items" : {}
        },
        "meta" : {
            "description" :  "Statistics" ,
            "type" :  "object" ,
            "properties" : {
                "CHANGE_DATE" : {
                    "description" :  "Modification date" ,
                    "type" :  "object" ,
                    "properties" : {
                        "from" : {
                            "description" :  "The first modification date" ,
                            "type" :  "string"
                        },
                        "to" : {
                            "description" :  "The last modification date" ,
                            "type" :  "string"
                        }
                    }
                },
                "totalCount" : {
                    "description" :  "Total number of items" ,
                    "type" :  "integer"
                }
            }
```

```
        }
    }
}
```

Example

```
GET /xapi/workstation/stat?filter[change_date][to]= 1494951762 &filter[change_date][from]= 1494951000
```

Response

```
{
   "data" : [],
   "meta" : {
     "CHANGE_DATE" : {
       "from" :  "2017-03-24 10:40:03" ,
       "to" :  "2017-05-24 12:10:22"
     },
     "totalCount" : 539693
   }
}
```

Getting events

The chapter contains the following information:

- Getting text extracted from multiple events;
- Getting extracted text of event;
- Getting event metadata;
- Getting multiple event contents;
- Getting info on events contents;
- Getting intercepted file of event;
- Getting the list of protected objects;
- Getting the list of titles;
- Getting list of security objects catalogues;
- Getting the list of policies;
- Getting the List of Events;
- Getting tags list;
- Getting statistics on events;
- Getting verdict;
- Changing events attributes.

Getting text extracted from multiple events

Method: `POST`

Resource: `event/texts`

Parameters (passed in the form of JSON):

- `response_type` – the response type:
  - `chunked` — `Transfer-Encoding: chunked` is used, where each text is passed in its own block and in the order which is specified in the request body; if there is no text, then null character "\0" is passed instead;

    - `multipart/mixed` – `multipart/mixed` rfc tech-ms format is used. This format is not part of the HTTP standard, however, it is used in emails, so it can be easily implemented. Besides, it is very similar to `multipart/form-data`. Each text is passed in its block and is separately marked with X-Event-ID title indicating the event the text belongs to. If there is no text, then it is either not passed or the text body is missing.
- `event` – an array of objects with the following fields:
    - `id` – event ID;
    - `tbsId` – tablespace with the event.

Examples:

`chunked`:

Query

```
POST /xapi/event/texts
X-API-Version: 1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
(empty line)
{"response_type": "chunked", "events": [{"id": 2, "tbsId": 4}, {"id": 4, "tbsId": 4}]}
```

Response

```
HTTP/1.1 200 OK
Content-Type: text/plain
Transfer-Encoding: chunked
(empty line)
1
\0 (no-text character for event 2)
48AC
(text content for event 4)
0
```

`multipart/mixed`:

Query

```
POST /xapi/event/texts
X-API-Version: 1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
(empty line)
{"response_type": "multipart/mixed", "events": [{"id": 2, "tbsId": 4}, {"id": 4, "tbsId": 4}]}
```

Response

```
HTTP/1.1 200 OK
Content-Type: multipart/mixed; boundary=Asrf456BGe4h
Content-Length: (total volume including titles, can be missing!)
(empty line)
(missing preamble)
```

```
--Asrf456BGe4h
Content-Type: application/x-empty
Content-Disposition: attachment; filename="text-2.txt"
X-Event-ID: 2
(empty line and zero content then)
--Asrf456BGe4h
Content-Disposition: attachment; filename="text-4.txt"
Content-Type: text/plain
X-Event-ID: 4
(empty line)
(event content for event 4)
--Asrf456BGe4h--
(missing epilogue - empty line)
```

Getting extracted text of event

Method: `POST`

Resource: `event/texts`

Parameters (transferred as JSON):

- `response_type` – type of response:
  - `chunked` – using `Transfer-Encoding: chunked`, where each text is left within its block and order, transferred within the query body. In case there is no text, null, symbol "\0" is transferred instead of the text;
  - `multipart/mixed` – `multipart/mixed` rfc tech-ms format is used, it is not a HTTP format, but it is used in e-mails. That's why this can be realised. It is also similar to `multipart/form-data`. Each text is left within its block and is specified by X-Event-ID title in accordance with event it belongs to. In case there is no text, it cannot be transferred or the text body is absent;
- `events` – objects massive with fields:
  - `id` – event ID;
  - `tbsId` – event tablespace.

Examples:

`chunked`:

Query

```
 POST /xapi/event/texts
X-API-Version: 1.1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
(empty string)
{"response_type": "chunked", "events": [{"id": 2, "tbsId": 4}, {"id": 4, "tbsId": 4}]}
```

Response

```
HTTP/1.1 200 OK
Content-Type: text/plain
Transfer-Encoding: chunked
(empty string)
```

```
1
\0 (text absence symbol for event 2)
48AC
(text content for event 4)
0
```

`multipart/mixed` :

Query

```
POST /xapi/event/texts
X-API-Version: 1.1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
(empty string)
{"response_type": "multipart/mixed", "events": [{"id": 2, "tbsId": 4}, {"id": 4, "tbsId": 4}]}
```

Response

```
HTTP/1.1 200 OK
Content-Type: multipart/mixed; boundary=Asrf456BGe4h
Content-Length: (summary amount, including child titles can be absent!)
(empty string)
(absent preamble)
--Asrf456BGe4h
Content-Type: application/x-empty
Content-Disposition: attachment; filename="text-2.txt"
X-Event-ID: 2
(empty string and null content after)
--Asrf456BGe4h
Content-Disposition: attachment; filename="text-4.txt"
Content-Type: text/plain
X-Event-ID: 4
(empty string)
(text content for event 4)
--Asrf456BGe4h--
(absent epilogue - empty string)
```

Getting event metadata

Method: `GET`

Resource: `event/{id}`

Parameters:

- `{id}` – `OBJECT_ID` of event;
- `with[]` – list of additional entities, to be added to the object (for example, senders, recipients, etc.).

Response: Event

Example:

Get event OBJECT_ID "1" with senders and recipients

```
GET /xapi/event/1?with[]=senders&with[]=recipients
X-API-Version: 1
X-API-CompanyId: iw
X-API-ImporterName: aplab
X-API-Auth-Token: gla20dos1pcdbr69caxz
```

Getting multiple event contents

Method: `POST`

Resource: `event/raws`

Parameters (passed in the form of JSON):

- `response_type` – the response type:
  - `chunked` – `Transfer-Encoding: chunked` is used, where each content is passed in its own block and in the order which is specified in the request body; if there is no content, then null character " `\0` " is passed instead;
  - `multipart/mixed` – `multipart/mixed` rfc tech-ms format is used. This format is not part of the HTTP standard, however, it is used in emails, so it can be easily implemented. Besides, it is very similar to `multipart/form-data`. Each content is passed in its block and is separately marked with `X-Event-ID`, `X-Content-ID` and `X-Content-Deleted` titles which indicate the event and the content the block belongs to. If there is no content, then it is either not passed or the text body is missing.
- `event` – an array of objects with the following fields:
  - `id` – event ID;
  - `tbsId` – tablespace with the event;
  - `contentId` – content ID. If not specified, the main node is passed.

Examples:

`chunked` :

Query

```
POST /xapi/event/raws
X-API-Version: 1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
(empty line)
{"response_type": "chunked", "events": [{"id": 2, "tbsId": 4}, {"id": 3, "tbsId": 4}, {"id": 4, "tbsId": 4,
"contentId": 8}]}
```

Response

```
HTTP/1.1 200 OK
Content-Type: text/plain
Transfer-Encoding: chunked
(empty line)
1
\0 (no-content character for event 2)
3E4
```

```
(root content data for event 3)
48AC
(data of content 8 for event 4)
0
```

`multipart/mixed` :

Query

```
POST /xapi/event/raws
X-API-Version: 1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
(empty line)
{"response_type": "multipart/mixed", "events": [{"id": 2, "tbsId": 4}, {"id": 3, "tbsId": 4}, {"id": 4,
"tbsId": 4, "contentId": 8}]}
```

Response

```
HTTP/1.1 200 OK
Content-Type: multipart/mixed; boundary=Asrf456BGe4h
Content-Length: (total volume including titles, can be missing!)
(empty line)
(missing preamble)
--Asrf456BGe4h
Content-Type: application/x-empty
Content-Disposition: attachment; filename="content-2"
X-Event-ID: 2
(empty line and then zero content)
--Asrf456BGe4h
Content-Disposition: attachment; filename="content-3-5.eml"
Content-Type: email
X-Event-ID: 3
X-Content-ID: 5
(empty line)
(root content data for event 3)
--Asrf456BGe4h
Content-Disposition: attachment; filename="my-project.jpg"
Content-Type: image/jpeg
X-Event-ID: 4
X-Content-ID: 8
(empty line)
(data of content 8 for event 4)
--Asrf456BGe4h--

(missing epilogue - empty line)
```

Getting info on events contents

Method: `POST`

Resource: `event/contents`

Parameters (to be sent via JSON):

- `response_type` – types of response are as follows:
  - `chunked` – `Transfer-Encoding: chunked` is used, where each content is given in its block and in the order according to query body, in case content is absent, null `"\0"` character is given instead of content;
  - `multipart/mixed` – `multipart/mixed` rfc tech-ms format is used, it's not a part of HTTP standard but it is used in e-mails. Thus this can be easily realized. It also looks like `multipart/form-data`. Each content is given in its block and marked with `X-Event-ID` and `X-Content-ID` headers, to show which event and content it belongs to. In case content is absent, this will not be given or there is no text body;
- `events` – events array with fields as follows:
  - `id` – one or few events ID, used to get contents;
  - `tbsId` – tablespaces ID for each event;
- `filter[mime]`– content types where few values (exact match or mask) or all of them can be set;
- `start` – starting from which position the contents are to be given, by default is 0;
- `limit` – the number of contents to be given. By default is 100, cannot exceed 1000 in one query.

Example:

Query

```
curl -X POST "server.company.com/xapi/event/contents?filter[mime][]=image%2Fjpeg&filter[mime]
[]=text%2F%2A&start=1&limit=10" -H "accept: application/json" -H "Content-Type: application/json" -H "X-
Enable-OpenAPI: 1" -d "{\"events\":[{\"id\":2,\"tbsId\":4}]}"
```

By request there will be given content IDs which are in accordance with query conditions and containing the following attributes:

Response

```
{
  "data": [
    {
      "OBJECT_CONTENT_ID": 15,
      "MIME": "string",
      "OBJECT_ID": 2,
      "TBS_ID": 4,
      "CONTENT_SIZE": 1500,
      "is_filename": 0           // indicates the content is a part or not a part of filename
    }
  ],
  "meta": {
    "totalCount": 1,
    "start": 1,
    "limit": 10
  }
}
```

Getting intercepted file of event

Method: `GET`

Resource: `event/{id}/raw?download=0`

Parameters:

- `{id}` – OBJECT_ID of event, mandatory;
- `download` – adding HTTP-heading to download a file (`Content-Disposition` , `Content-Transfer-Encoding` and others), mandatory, the value is 1 or 0 (default value is 0).

Supporting HTTP-heading `Range` .

Response: File of intercepted event. For example, eml (for emails) or the file itself for copying file event.

Getting the list of protected objects

Description: The list of protected objects sorted by name.

Method: `GET`

Resource: `event/protectedDocument`

Parameters:

- `start` – a position from which to start returning protected objects, the default value is 0;
- `limit` – the number of protected objects to be returned; the default value is 100; cannot be more than 1000;
- `with[]` – the list of additional entities to be added to the protected object:
  - `pd2pc` – a link of the protected object with the catalog of protected objects.

Response: The list of protected objects described with the following JSON Schema:

JSON schema of returning the list of protected objects

```
{
     "$schema" :  "http://json-schema.org/draft-04/schema#" ,
     "type" :  "object" ,
     "properties" : {
         "data" : {
             "type" :  "array" ,
             "description" :  "Array of protected objects" ,
             "items" : {
                 "type" :  "object" ,
                 "description" :  "Protected object" ,
                 "properties" : {
                     "DOCUMENT_ID" : {
                         "type" :  "string" ,
                         "description" :  "Identifier"
                     },
                     "DISPLAY_NAME" : {
                         "type" :  "string" ,
                         "description" :  "Name"
                     },
                     "CREATE_DATE" : {
                         "type" :  "string" ,
                         "description" :  "Creation date"
                     },
                     "IS_DELETED" : {
                         "type" :  "integer" ,
                         "description" :  "Whether it is removed from the active configuration" ,
                         "enum" : [
                             0,
                             1
                         ]
                     },
```

```
                    "pd2pc" : {
                        "description" :  "Array of links to catalogs of protected objects" ,
                        "type" :  "array" ,
                        "items" : {
                            "description" :  "Link to the catalog of protected objects" ,
                            "type" :  "object" ,
                            "required" : [
                                "CATALOG_ID" ,
                                "DOCUMENT_ID"
                            ],
                            "properties" : {
                                "CATALOG_ID" : {
                                    "description" :  "A unique identifier of the catalog of protected
objects" ,
                                    "type" :  "string"
                                },
                                "DOCUMENT_ID" : {
                                    "description" :  "A unique identifier of the protected object" ,
                                    "type" :  "string"
                                }
                            }
                        }
                    }
                },
                "required" : [
                    "DOCUMENT_ID" ,
                    "DISPLAY_NAME" ,
                    "CREATE_DATE" ,
                    "IS_DELETED"
                ]
            }
        },
        "meta" : {
            "description" :  "Additional data" ,
            "type" :  "object" ,
            "properties" : {
                "totalCount" : {
                    "type" :  "integer" ,
                    "description" :  "The total number of protected objects"
                },
                "start" : {
                    "type" :  "integer" ,
                    "description" :  "The position from which returning has started"
                },
                "limit" : {
                    "type" :  "integer" ,
                    "description" :  "The number of returned objects"
                }
            },
            "required" : [
                "totalCount" ,
                "start" ,
                "limit"
            ]
        }
    },
    "required" : [
        "data" ,
```

```
        "meta"
    ]
}
```

Example:

Query

```
GET /xapi/event/protectedDocument?start= 0 &limit= 10
X-API-Version:  1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
```

Response

```
{
    "data" : [{
        "DOCUMENT_ID" :  "C78E821A67C7F34A1B1D63A4151CEF6300000000" ,
        "DISPLAY_NAME" :  "Security labels" ,
        "CREATE_DATE" :  "2016-05-12 10:24:41.342554" ,
        "IS_DELETED" : 0
    }],
    "meta" : {
        "totalCount" : 1,
        "start" : 0,
        "limit" : 10
    }
}
```

Getting the list of titles

Description: Getting the list of additional fields of the event

Method: `GET`

Resource: `event/header`

Parameters:

- `start` – the start position for returning titles; the default value is 0;
- `limit` – the number of titles to be returned; the default value is 100; cannot be more than 1000.

Response: The list of additional fields described with the following **JSON return schema:**

```
{

    "$id": "http://infowatch.com/iwtm/xapi/event/header.json",

    "type": "object",

    "$schema": "http://json-schema.org/draft-07/schema#",

    "properties": {
```

```
        "data": {
            "type": "array",
            "items": {
                "type": "object",
                "properties": {
                    "NAME": {
                        "type": "string",
                        "title": "Attribute code that will be used in the event titles and for
identification in xAPI/PushAPI",
                        "examples": [
                            "mac_level"
                        ]
                    },
                    "NOTE": {
                        "type": "string",
                        "title": "Attribute name which is displayed in TM Console",
                        "examples": [
                            "Mandatory access level"
                        ]
                    },
                    "TYPE": {
                        "type": "string",
                        "title": "One of the following values: number (integer, fractional, string), date
and time in UTC format + specifying time zone offset, duration, hyperlink",
                        "enum": [
                            "integer",
                            "float",
                            "string",
                            "date",
                            "UTC",
                            "duration",
```

```
                            "link"
                        ]
                    },
                    "IS_DELETED": {
                        "type": "integer",
                        "title": "Whether the title is removed from the System (for obsolete objects, it
remains)",
                        "default": 0,
                        "enum": [
                            0,
                            1
                        ]
                    },
                    "FORMAT": {
                        "type": "string",
                        "title": "Value type (only for version 1.2)",
                        "enum": [
                            "string",
                            "date",
                            "number"
                        ]
                    },
                    "IS_SYSTEM": {
                        "type": "integer",
                        "title": "Whether the title is system (only for version 1.1)",
                        "default": 1,
                        "enum": [
                            0,
                            1
                        ]
```

```
                    },
                    "IS_MULTIPLE_VALUE": {
                        "type": "integer",
                        "title": "Can the title contain several values (for verrion 1.1)",
                        "default": 0,
                        "enum": [
                            0,
                            1
                        ]
                    },
                    "locale": {
                        "type": "array",
                        "title": "Titles localization (for version 1.2)",
                        "items": {
                            "type": "object",
                            "title": "Title localization",
                            "properties": {
                                "NAME": {
                                    "type": "string",
                                    "title": "Attribute code that will be used in the event titles and for
identification in xAPI/PushAPI"
                                },
                                "LANGUAGE": {
                                    "type": "string",
                                    "title": "Language name",
                                    "enum": [
                                        "rus",
                                        "eng"
                                    ]
                                },
```

```
                        "DISPLAY_NAME": {
                            "type": "string",
                            "title": "Attribute name which is displayed in TM Console"
                        }
                    }
                }
            }
        }
    },
    "meta": {
        "description": "Additional data",
        "type": "object",
        "properties": {
            "totalCount": {
                "type": "integer",
                "description": "Total number of protected objects"
            },
            "start": {
                "type": "integer",
                "description": "From which position has it started to return"
            },
            "limit": {
                "type": "integer",
                "description": "Total number of returned"
            }
        },
        "required": [
            "totalCount",
```

```
                "start",
                "limit"
            ]
        }
    },
    "required": [
        "data",
        "meta"
    ]
}
```

Example:

Query

```
GET /xapi/event/header
X-API-Version:  1.1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
```

Response

```
{
    "data": [
        {
            "NAME": "device_type",
            "NOTE": "Device type",
            "TYPE": "enum",
            "IS_DELETED": 0,
            "IS_SYSTEM": 1,
            "IS_MULTIPLE_VALUE": 0
        },
        {
            "NAME": "destination_type",
```

```
        "NOTE": "Type of the scanning source",
        "TYPE": "enum",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "destination_url",
        "NOTE": "URL",
        "TYPE": "string",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "print_server",
        "NOTE": "Print server name",
        "TYPE": "string",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "location",
        "NOTE": "Printer location",
        "TYPE": "string",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
```

```
    },
    {
        "NAME": "comment",
        "NOTE": "Comment to the printer",
        "TYPE": "string",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "port_name",
        "NOTE": "Name of the printer port",
        "TYPE": "string",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "copies",
        "NOTE": "The number of copies",
        "TYPE": "number",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "task_name",
        "NOTE": "Task name",
        "TYPE": "string",
```

```
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "print_task_name",
        "NOTE": "Print task name",
        "TYPE": "string",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "task_run_date",
        "NOTE": "Task start date",
        "TYPE": "date",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "create_date",
        "NOTE": "File creation date",
        "TYPE": "date",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
```

```
        "NAME": "modify_date",
        "NOTE": "File modification date",
        "TYPE": "date",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "source_ip",
        "NOTE": "IP address of the source",
        "TYPE": "string",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "workstation_type",
        "NOTE": "Workstation type",
        "TYPE": "enum",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "application_path_from",
        "NOTE": "Path to the application from which the event has been copied",
        "TYPE": "string",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
```

```
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "application_path_to",
        "NOTE": "Path to the application, to which the event has been copied",
        "TYPE": "string",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "terminal_session",
        "NOTE": "Terminal session indicator",
        "TYPE": "number",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "emitter",
        "NOTE": "Event source",
        "TYPE": "enum",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1,
        "IS_MULTIPLE_VALUE": 0
    },
    {
        "NAME": "charset",
        "NOTE": "Encoding of the attachment text",
```

```
            "TYPE": "string",

            "IS_DELETED": 0,

            "IS_SYSTEM": 1,

            "IS_MULTIPLE_VALUE": 0

        }

    ],

    "meta": {

        "totalCount": 20,

        "start": 0,

        "limit": 100

    }

}
```

Getting list of security objects catalogues

Description: List of security objects catalogues, sorted out by name

Method: `GET`

Resource: `event/protectedCatalog`

Parameters:

- `start` – from what position the catalogue should be started to be given (default value is 0);
- `limit` – how many catalogues are to be given (default value is 100, but it can't be more than 1000).

Response: List of security objects catalogues, described in **the JSON-Schema** below:

JSON scheme of catalogues list output

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "properties": {
        "data": {
            "type": "array",
            "description": "Массив каталогов ОЗ",
            "items": {
                "type": "object",
                "description": "Каталог ОЗ",
                "properties": {
                    "CATALOG_ID": {
                        "type": "string",
                        "description": "Идентификатор"
                    },
                    "DISPLAY_NAME": {
```

```
                        "type": "string",
                        "description": "Имя"
                    },
                    "CREATE_DATE": {
                        "type": "string",
                        "description": "Дата создания"
                    },
                    "IS_DELETED": {
                        "type": "integer",
                        "description": "Удален ли из активной конфигурации",
                        "enum": [
                            0,
                            1
                        ]
                    }
                },
                "required": [
                    "CATALOG_ID",
                    "DISPLAY_NAME",
                    "CREATE_DATE",
                    "IS_DELETED"
                ]
            }
        },
        "meta": {
            "description": "Дополнительные данные",
            "type": "object",
            "properties": {
                "totalCount": {
                    "type": "integer",
                    "description": "Общее количество ОЗ"
                },
                "start": {
                    "type": "integer",
                    "description": "С какой позиции началось отдаваться"
                },
                "limit": {
                    "type": "integer",
                    "description": "Сколько всего отдалось"
                }
            },
            "required": [
                "totalCount",
                "start",
                "limit"
            ]
        }
    },
    "required": [
        "data",
        "meta"
    ]
}
```

Example:

Request

```
GET /xapi/event/protectedCatalog?start=0&limit=10
X-API-Version: 1
X-API-CompanyId: iw
X-API-ImporterName: aplab
X-API-Auth-Token: gla20dos1pcdbr69caxz
```

Response

```
{
    "data": [{
        "CATALOG_ID": "3D0F8CDAF9E74F82AC85EB32D12727A700000000",
        "DISPLAY_NAME": "Грифованная информация",
        "CREATE_DATE": "2016-05-12 10:24:41.342554",
        "IS_DELETED": 0
    }],
    "meta": {
        "totalCount": 1,
        "start": 0,
        "limit": 10
    }
}
```

Getting the list of policies

escription: The list of policies sorted by name

Method: `GET`

Resource: `event/policy`

Parameters:

- `start` – a position from which to start returning policies, the default value is 0;
- `limit` – the number of policies to be returned; the default value is 100; cannot be more than 1000.

Response: The list of policies described with the following JSON Schema:

JSON schema of returing the list of policies

```
{
     "$schema" :  "http://json-schema.org/draft-04/schema#" ,
     "type" :  "object" ,
     "properties" : {
         "data" : {
             "type" :  "array" ,
             "description" :  "Array of policies" ,
             "items" : {
                 "type" :  "object" ,
                 "description" :  "Policy" ,
                 "properties" : {
                     "POLICY_ID" : {
                         "type" :  "string" ,
                         "description" :  "Identifier"
```

```
                    },
                    "DISPLAY_NAME" : {
                        "type" :  "string" ,
                        "description" :  "Name"
                    },
                    "CREATE_DATE" : {
                        "type" :  "string" ,
                        "description" :  "Creation date"
                    },
                    "IS_DELETED" : {
                        "type" :  "integer" ,
                        "description" :  "Whether it is removed from the active configuration" ,
                        "enum" : [
                            0,
                            1
                        ]
                    }
                },
                "required" : [
                    "POLICY_ID" ,
                    "DISPLAY_NAME" ,
                    "CREATE_DATE" ,
                    "IS_DELETED"
                ]
            }
        },
        "meta" : {
            "description" :  "Additional data" ,
            "type" :  "object" ,
            "properties" : {
                "totalCount" : {
                    "type" :  "integer" ,
                    "description" :  "Total number of policies"
                },
                "start" : {
                    "type" :  "integer" ,
                    "description" :  "A position where to start returning policies"
                },
                "limit" : {
                    "type" :  "integer" ,
                    "description" :  "Total number of returned policies"
                }
            },
            "required" : [
                "totalCount" ,
                "start" ,
                "limit"
            ]
        }
    },
    "required" : [
        "data" ,
        "meta"
    ]
}
```


Example:

Query

```
GET /xapi/event/policy?start= 0 &limit= 10
X-API-Version:  1
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
```

Response

```
{
     "data" : [{
         "POLICY_ID" :  "3D0F8CDAF9E74F82AC85EB32D12727A700000000" ,
         "DISPLAY_NAME" :  "Security labels" ,
         "CREATE_DATE" :  "2016-05-12 10:24:41.342554" ,
         "IS_DELETED" : 0
     }],
     "meta" : {
         "totalCount" : 1,
         "start" : 0,
         "limit" : 10
     }
}
```

Getting the List of Events

Description: The list of events sorted out by the time of interception (from new to old ones)

Method: `GET`

Resource: `event`

Parameters:

- `filter` – the filter list:
  - `date[from]` – start date in UNIX-timestamp format (can be used separately from `date[to]`);
  - `date[to]` – end date in UNIX-timestamp format (can be used separately from `date[from]`);
  - `insert_date[>]` – start insert date in UNIX-timestamp format separated by microtime dot, for example, `filter[insert_date][>]=1234.123312`;
  - `insert_date[<]` – end insert date in UNIX-timestamp separated by microtime dot, for example, `filter[insert_date][<]=1234.123312`;
  - `insert_date[>=]` – start insert date in UNIX-timestamp format separated by microtime dot, for example `filter[insert_date][>=]=1234.123312` (incl.) (API 1.1 and later versions);
  - `insert_date[<=]` – end insert date in UNIX-timestamp separated by microtime dot, for example `filter[insert_date][<=]=1234.123312` (incl.) (API 1.1 and later versions);
  - `rule_group_type[]` – rule group (API 1.1 and later versions);
  - `violation_level[]` – violation level (API 1.1 and later versions);
  - `verdict[]` – verdict (API 1.1 and later versions);
  - `object_type_code[]` – type of event (can be several values);
  - `protocol[]` – protocol (can be several values);
  - `user_decision[]` – user decision (can be several values);

    - `object_size[from]` – the minimum number of bytes for event search (can be used separately from `object_size [to]`);
    - `object_size[to]` – the maximum number of bytes for event search (can be used separately from `object_size [from]`);
    - `protected_catalog_id[]` – an ID of the catalog with protected objects (can be several values);
    - `tag_id[]` – tag ID (can be several values);
- `with[]` – the list of additional entities to be added to the object (for example, senders, recipients, etc.):
        - `protected_catalogs`;
        - `protected_documents`;
        - `policies`;
        - `senders_keys`;
        - `senders`;
        - `recipients_keys`;
        - `recipients`;
        - `workstations_keys`;
        - `workstations`;
        - `tags`;
        - `perimeters` (API 1.1 and later versions);
        - `attachments` (API 1.1 and later versions);
        - `files` (API 1.1 and later versions) – an undocumented parameter which returns all contents with unempty *FILE_NAME* field;
        - `contents_renderer` (API 1.8 and later versions) – attach renderer contents with different type contents.
    - `without[]` – additional flags:
        - `count` – omit calculating `totalCount` (calculating `totalCount` may take more than halftime of the query, so we recommend using `event/stat` method instead).
    - `headers[]` – the list of additional fields (titles) of the event (for all fields, you * character). You can get the list with `event/header` method (API 1.1 and later versions);
    - `sort[]` – an undocumented field for sorting the result (significantly slows down getting events; for getting the first/last event, we recommend using `event/stat` method. Possible keys:
        - `capture_date` – `asc` (API 1.1 and later vesions) `/desc`, for example, `?sort[capture_date]=desc`;
        - `insert_date` – `asc/desc`, for example, `?sort[insert_date]=asc` (API 1.1 and later versions).
- `start` – a start position for returning events (the default value is 0);
- `limit` – how many events to return (the default value is 100, cannot be more than 1000).

Steps to take when receiving events for the first time:

1. Define the date starting from which it is required to receive events. For example, to receive an interval `INSERT_DATE` from the events statistics (`GET /xapi/event/stat`).
2. Divide the interval by the smallest value: number of streams (not higher than 32) **or** number of cores on server.
3. In each stream, create and run the first request to receive a list of event which will look as follows:
   `GET /xapi/event?without[]=count&filter[insert_date]`

`[>=]={interval_minimum}&filter[insert_date][<=]={interval_maximum}&sort[insert_date]=asc&limit=1000`

4. Remember all `OBJECT_ID` in the received list.
5. In the received list, find the last `INSERT_DATE` and use it as `interval_minimum` in the request described at step 3.
6. Exclude the events which were remembered in the last iteration, from the new list.
7. If no `interval_maximum`, start again from step 4.
8. Save to persistent storage:
    a. `INSERT_DATE` maximum value, received at step 1.
    b. `OBJECT_ID` list from the interval, which includes INSERT_DATE maximum value.

Examples:

Get the first 100 events of January 2015

```
GET /xapi/event?filter[date][from]=1420059600&filter[date][to]=1422737999&start=0&limit=100
X-API-Version: 1
X-API-CompanyId: iw
X-API-ImporterName: aplab
X-API-Auth-Token: gla20dos1pcdbr69caxz
```

Get 1000 events with emails that contain content

```
GET /xapi/event?with=contents_renderer&limit=1000
X-API-Version: 1.8
X-API-CompanyId: IW
X-API-productName: Vision
X-API-Auth-Token: 1eja07v23vs9k1s5ij5k
```

The response contains the list of events sorted out by the time of interception (from new to old ones). For different event types, the response structure will be different.

Sample response

```
[
 OBJECT_ID: 1
 VIOLATION_LEVEL: 'High'
 CAPTURE_DATE: '12-03-2022 15:02:25'
 contents_renderer:
 [
    SUBJECT: [
       CONTENT_ID: 1
       OBJECT_ID: 1
       CONTENT_SIZE: 26
   ]
   BODY: [
      CONTENT_ID: 2
      OBJECT_ID: 1
      CONTENT_SIZE:5125
   ]
 ]
]
```

Getting tags list

Description: Tags list, sorted out by name.

Method: `GET`

Resource: `event/tag`

Parameters:

- `start` – from what position the catalogue should be started to be given (default value is 0);
- `limit` – how many catalogues are to be given (default value is 100, but it can't be more than 1000).

Response: List of tags, described in the *JSON-Schema* below:

JSON scheme of tags list output

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "properties": {
        "data": {
            "type": "array",
            "description": "Массив тегов",
            "items": {
                "type": "object",
                "description": "Тег",
                "properties": {
                    "TAG_ID": {
                        "type": "string",
                        "description": "Идентификатор"
                    },
                    "DISPLAY_NAME": {
                        "type": "string",
                        "description": "Имя"
                    },
                    "COLOR": {
                        "type": "string",
                        "description": "Цвет тега в HEX RGP, например #ffffff"
                    },
                    "CREATE_DATE": {
                        "type": "string",
                        "description": "Дата создания"
                    },
                    "IS_DELETED": {
                        "type": "integer",
                        "description": "Удален ли из активной конфигурации"
                    },
                    "IS_SYSTEM": {
                        "type": "integer",
                        "description": "Является ли тег системным"
                    }
                },
                "required": [
                    "TAG_ID",
                    "DISPLAY_NAME",
                    "COLOR",
```

```
                    "CREATE_DATE",
                    "IS_DELETED",
                    "IS_SYSTEM"
                ]
            }
        },
        "meta": {
            "description": "Дополнительные данные",
            "type": "object",
            "properties": {
                "totalCount": {
                    "type": "integer",
                    "description": "Общее количество ОЗ"
                },
                "start": {
                    "type": "integer",
                    "description": "С какой позиции началось отдаваться"
                },
                "limit": {
                    "type": "integer",
                    "description": "Сколько всего отдалось"
                }
            },
            "required": [
                "totalCount",
                "start",
                "limit"
            ]
        }
    },
    "required": [
        "data",
        "meta"
    ]
}
```

Example:

Request

```
GET /xapi/event/tag?start=0&limit=10
X-API-Version: 1
X-API-CompanyId: iw
X-API-ImporterName: aplab
X-API-Auth-Token: gla20dos1pcdbr69caxz
```

Response

```
{
    "data": [{
        "TAG_ID": "0AF69DD5742E6977E0530100007F2BB700000000",
        "DISPLAY_NAME": "VIP",
        "COLOR": "#4e954e",
        "CREATE_DATE": "2016-05-12 10:24:40.002673",
        "IS_DELETED": 0,
        "IS_SYSTEM": 1
```

```
    }],
    "meta": {
        "totalCount": 1,
        "start": 0,
        "limit": 10
    }
}
```

Getting statistics on events

Description: An undocumented method of getting statistics on events, for now, it only provides statistics by the interception date

Method: `GET`

Resource: `event/stat`

Parameters:

- `fields` – a list of fields to be selected: `CAPTURE_DATE`, `INSERT_DATE`, `totalCount` (API 1.2 version);
- `filter` – a filtering list:
    - `date[from]` – start date in the UNIX-timestamp format (can be used separately from `date[to]`);
    - `date[to]` – end date in the UNIX-timestamp format (can be used separately from `date[from]`);
    - `insert_date[>]` – date of start pasting in the UNIX-timestamp format and via microtime point, for example `filter[insert_date][>]=1234.123312` (starting from API 1.1 version);
    - `insert_date[<]` – date of end pasting in the UNIX-timestamp format and via microtime point, for example `filter[insert_date][>]=1234.123312` (starting from API 1.1 version);
    - `insert_date[>=]` – date of start pasting in the UNIX-timestamp format and via microtime point, for example `filter[insert_date][>]=1234.123312` (including) (starting from API 1.1 version);
    - `insert_date[<=]` – date of end pasting in the UNIX-timestamp format and via microtime point, for example `filter[insert_date][>]=1234.123312` (including) (starting from API 1.1 version);
    - `object_type_code[]` – type of event (could be few values) (starting from API 1.1 version);
    - `violation_level[]` – level of violation (starting from API 1.1 version).

Response: Statistics on events described with the following *JSON Schema*:

JSON return schema

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "properties": {
        "data": {
            "description": "Empty section",
            "items": {},
            "type": "array"
        },
```

```
        "meta": {
            "description": "Additional data",
            "type": "object",
            "properties": {
                "CAPTURE_DATE": {
                    "description": "Statistics by field CAPTURE_DATE",
                    "type": "object",
                    "properties": {
                        "from": {
                            "type": "string",
                            "description": "The date of capture of the first event in selection"
                        },
                        "to": {
                            "type": "string",
                            "description": "The date of capture of the last event in selection"
                        }
                    }
                },
                "INSERT_DATE": {
                    "description": "Statistics by field INSERT_DATE (API 1.1 version)",
                    "type": "object",
                    "properties": {
                        "from": {
                            "type": "string",
                            "description": "The date of insertion of the first event in selection"
                        },
                        "to": {
                            "type": "string",
                            "description": "The date of insertion of the last event in selection"
                        }
                    }
                },
                "totalCount": {
                    "type": "integer",
                    "description": "Total number of events"
                }
            }
        }
    }
}
```

Example:

Query

```
GET /xapi/event/stat?filter[data][from]= 0
X-API-Version:  1.3
X-API-CompanyId: iw
X-API-ImporterName: dome
X-API-Auth-Token: brse99dugp1cdbr69axz
```

Response

```
{
    "data": [],
    "meta": {
```

```
        "CAPTURE_DATE": {
            "from": "2017-04-16 00:00:04",
            "to": "2017-05-16 16:20:36"
        },
        "INSERT_DATE": {
            "from": "2017-04-16 00:01:07",
            "to": "2017-05-16 16:20:40"
        },
        "totalCount": 225075
    }
}
```

Getting verdict

API Version: 1.8 and later

Method: `GET`

Resource: `/event/verdict`

Description:

The method returns an event verdict based on the document ID:

- `Allowed`
- `Quarantined`
- `Forbidden`

Besides, the method's output includes the following fields:

- `OBJECT_ID`, `integer` – unique object ID
- `TBS_ID`, `number` – tablespace ID for partitioning

Parameters:

- `documentId` – ID of the intercepted document
- `date` – interception date in the *YYYY-MM-DD* format (UTC+0:00)

Example:

Query

```
curl -X 'GET' \
  'https://server.company.ru/xapi/event/verdict?documentId=b042cae7-
cd7f-4ff3-9af7-5ed41ec0dc23&date=2024-01-15' \
  -H 'X-API-Version: 1.8'
```

Response

```
{
  "data": [
    {
      "VERDICT": "Allowed",
      "DOCUMENT_ID": "string",
      "OBJECT_ID": 0,
      "TBS_ID": 0
    }
  ]
}
```

To get event verdicts fast, we recommend making a query for several `document ID` at once:

Query

```
https://server.infowatch.ru/xapi/event/verdict?documentId=ID1,ID2,ID3,ID4,ID5,ID6,IDnnn&date=DATE2'
```

Tracking the verdict through the document

If you need to check a set of files from an external system and take some actions when the policies are triggered and the user has made a decision on the event, then you need to:

1. Send the files to TM for checking
2. Get and store the information about the TM event
3. Track the event status

Sending files to TM for checking
The system stores the information about the process of file checking. It has the following structure:

- `file_path` – path or the full name of the file to be sent to analysis
- `document_id` – event GUID, generated when the file is sent to analysis
- `object_id` – identifier of the event that was created in TM as a result of the analysis
- `tbs_id` – table space identifier
- `verdict` – verdict made as a result of the analysis
- `user_decision` – user decision after placing the event to quarantine

The file is sent via a PushAPI query:

Example of query for sending a file to TM for analysis

```
pushapi-util --host 127.0.0.1 --token 8iwivt0cmirlcceuu12l --id IW --class fileexch --service im_vk --
evtattr event_name:"vkontakte Event" --sender "vkontakte:sender1111:'{\"display_name\":\"Vrbt tester\",
\"url\":\"https://www.facebook.com/100006160150433\",\"login\":\"new_login\"}'" --receiver
vkontakte:vkontaktereceiver1 --receiver "vkontakte:11111123:'{\"url\":\"https://vk.com/vrbttester\",
\"login\":\"some_login\"}'" --data file:file.txt,filename:file.txt
```

Example of response with document_id of the event

```
Object database GUID: df5f046e-0555-46e1-be48-8da33f41b9bb
```

You need to repeat the operation for each file and save all the received GUIDs. They are to be used to track the analysis status and to change the user decision.

Getting events from TM
This action is required to get the information about triggered policies and the verdict made for the TM event after the analysis of the file content. Furthermore, the step requires receiving the event identifiers `OBJECT_ID` and `TBS_ID` based on their GUID (`document_id`).

For this, you need to make a Public API query as follows:

Example of query

```
https://server.infowatch.ru/xapi/event?filter[document_id]={GUID_1},{GUID_2},...,{GUID_N}&filter[date]
[from]={START_TIME}&filter[date][to]={END_TIME}
```

where `{GUID_i}` is the GUID (`document_id`) of the file received when sending the file to analysis.

The result of the query is an array consisting of TM events. By matching the `document_id` stored as a result of sending files to analysis with `document_id` of the event received at this step, `OBJECT_ID` and `TBS_ID` are stored, as well as the parameters required for making further decision regarding the files being checked (for example, a verdict). Verdicts by `document_id` will be received for documents that are in the database at the time of the query. If there is no document in the database for the queried `document_id` , then the verdict for such a document will not be received.

A certain amount of time is needed to process events in TM. So, it is possible that the result of the API query does not contain the information about all the events for which `document_id` was transferred. Hence, you need to complete the public API query again for the `document_id` , for which the events were not received, until all of them are processed.

Tracking event status
To track the changes in user decision, we recommend periodically making audit queries for events where the decision was changed. The query may look like this:

Example of query
`https://server.infowatch.ru/xapi/audit?filter[change_date][>]={START_TIME}&filter[entity_type][]=Object`

where `{START_TIME}` is the Unix timestamp of the moment after the last audit query (for the first iteration – the moment after sending the first file to analysis).
After completing the query, all the audit events are processed one by one, the `ENTITY_ID` field in them is matched with the previously stored `OBJECT_ID` . The decision can be made based on the `OPERATION` and `PROPERTY_CHANGES` fields.

Changing events attributes

The chapter contains the following information:

- Assigning user verdict;
- Assigning user verdict to multiple events.

Assigning user verdict

API version: ≥ 1.2

Method: `PATCH`

Resource: `event/{id}/userDecision?tbsId={tbsId}`

Description:

The method is used to assign the user decision. The fields as follows:

- `{id}` – event ID (`OBJECT_ID`);
- `{tbsId}` – Event tablespace(`TBS_ID`);
- `USER_DECISION` – User decision. Possible values:
  - `NotProcessed` – no decision made;
  - `Violation` – there was a violation;
  - `NoViolation` – there was no violation;
  - `AdditionalProcessingNeeded` – additional processing is required.
- `NOTE` – a comment to be saved within **Audit** section. It may contain for example, a username of account from third company system, who has changed the attribute's value. The filed's length should not exceed 1000 characters.

Calling of this API is recorded into **Audit** section. When assigning a user decision the following fields could be automatically changed within the event:

- `TBS_ID`;
- `VERDICT`;
- `FORWARD_STATE_CODE`.

That's why in API respond there are the values that could be updated within your copy of event.

Query body: JSON scheme for user verdict assignment

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "type": "object",
  "title": "Set user decision schema",
  "required": [
    "USER_DECISION"
  ],
  "properties": {
    "USER_DECISION": {
      "$id": "#/properties/USER_DECISION",
      "type": "string",
      "title": "User decision",
      "enum": [
        "NotProcessed",
        "Violation",
        "NoViolation",
        "AdditionalProcessingNeeded"
      ]
    },
    "NOTE": {
      "$id": "#/properties/NOTE",
      "type": "string",
      "title": "Note",
      "maxLength": 1000
    }
  }
}
```

Response body: JSON scheme for user verdict assignment

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "type": "object",
  "title": "Result setting user decision schema",
  "required": [
    "OBJECT_ID",
    "TBS_ID",
    "VERDICT",
    "USER_DECISION",
    "FORWARD_STATE_CODE"
  ],
  "properties": {
    "OBJECT_ID": {
      "$id": "#/properties/OBJECT_ID",
      "type": "integer",
      "title": "Event ID"
```

```
    },
    "TBS_ID": {
      "$id": "#/properties/TBS_ID",
      "type": "integer",
      "title": "Tablespace ID"
    },
    "VERDICT": {
      "$id": "#/properties/VERDICT",
      "type": "string",
      "title": "Вердикт",
      "enum": [
        "Allowed",
        "Quarantined",
        "Forbidden"
      ]
    },
    "USER_DECISION": {
      "$id": "#/properties/USER_DECISION",
      "type": "string",
      "title": "Assigned user decision",
      "enum": [
        "NotProcessed",
        "NoViolation",
        "Violation",
        "AdditionalProcessingNeeded"
      ]
    },
    "FORWARD_STATE_CODE": {
      "$id": "#/properties/FORWARD_STATE_CODE",
      "type": "string",
      "title": "Sending status",
      "examples": [
        "Sent",
        "NotSent",
        "Pending"
      ]
    }
  }
}
```

Example: Assigning **Violation** verdict containing comment to event ID 194 in TBS 1 by mens of Url:

```
curl 'https://example.com/xapi/event/195/userDecision?tbsId=1' \
  -X PATCH \
  -H 'X-API-Auth-Token: 1792jmclf7fer1ikuhby' \
  -H 'X-API-Version: 1' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-ImporterName: dome' \
  -H 'Content-Type: application/json' \
  --data-binary '{"USER_DECISION":"Violation","NOTE":"Assignign verdict via Dome. Employee john Smith."}'
```

Assigning user verdict to multiple events

API Version: ≥ 1.2

Method: `PATCH`

Resource: `event/userDecision`

Description:

The method is used to assign user decision to multiple events (no more than **1000** user decisions, otherwise **413** error message will be returned). Event ID and tablespace ID are sent within the request body containing user decision and comment.

Request body: JSON scheme for multiple assignment of verdicts

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "type": "array",
  "title": "Set batch user decision schema",
  "items": {
    "$id": "#/items",
    "type": "object",
    "required": [
      "OBJECT_ID",
      "TBS_ID",
      "USER_DECISION"
    ],
    "properties": {
      "OBJECT_ID": {
        "$id": "#/items/properties/OBJECT_ID",
        "type": "integer",
        "title": "Event ID"
      },
      "TBS_ID": {
        "$id": "#/items/properties/TBS_ID",
        "type": "integer",
        "title": "Tablespace ID"
      },
      "USER_DECISION": {
        "$id": "#/items/properties/USER_DECISION",
        "type": "string",
        "title": "User decision",
        "enum": [
          "NotProcessed",
          "Violation",
          "NoViolation",
          "AdditionalProcessingNeeded"
        ]
      },
      "NOTE": {
        "$id": "#/items/properties/NOTE",
        "type": "string",
        "title": "Note",
        "maxLength": 1000
      }
    }
  }
}
```

Response body: JSON scheme for multiple assignment of verdicts

```
 {
```

```
"$schema": "http://json-schema.org/draft-07/schema#",
"type": "array",
"title": "Array result",
"items": {
  "$id": "#/items",
  "type": "object",
  "title": "Result setting user decision schema",
  "required": [
    "OBJECT_ID",
    "TBS_ID",
    "VERDICT",
    "USER_DECISION",
    "FORWARD_STATE_CODE"
  ],
  "properties": {
    "OBJECT_ID": {
      "$id": "#/items/properties/OBJECT_ID",
      "type": "integer",
      "title": "Event ID"
    },
    "TBS_ID": {
      "$id": "#/items/properties/TBS_ID",
      "type": "integer",
      "title": "Tablespace ID"
    },
    "VERDICT": {
      "$id": "#/items/properties/VERDICT",
      "type": "string",
      "title": "Verdict",
      "enum": [
        "Allowed",
        "Quarantined",
        "Forbidden"
      ]
    },
    "USER_DECISION": {
      "$id": "#/items/properties/USER_DECISION",
      "type": "string",
      "title": "User decision assigned",
      "enum": [
        "NotProcessed",
        "NoViolation",
        "Violation",
        "AdditionalProcessingNeeded"
      ]
    },
    "FORWARD_STATE_CODE": {
      "$id": "#/items/properties/FORWARD_STATE_CODE",
      "type": "string",
      "title": "Sending status",
      "examples": [
        "Sent",
        "NotSent",
        "Pending"
      ]
    }
  }
}
```

```
}
```

Example

```
curl 'https://example.com/xapi/event/userDecision' \
  -X PATCH \
  -H 'X-API-Auth-Token: 1792jmclf7fer1ikuhby' \
  -H 'X-API-Version: 1' \
  -H 'X-API-CompanyId: IW' \
  -H 'X-API-ImporterName: dome' \
  -H 'Content-Type: application/json' \
  --data-binary '[{"OBJECT_ID": 1, "TBS_ID": 1, "USER_DECISION":"Violation","NOTE":"Verdict assigning via
Dome. Employee John Smith."},{"OBJECT_ID": 24, "TBS_ID": 2, "USER_DECISION":"AdditionalProcessingNeeded"}]'
```

Retrieving the list of Console users

Description: The list of Traffic Monitor Console users (including former employees)

Method: `GET`

Resource: `user`

Parameters (are passed in JSON format):

- `start` – a position from which user data should be delivered (the default value is 0);
- `limit` – how many user data sets should be delivered (the default value is 100, the value cannot be larger than 1000).

JSON scheme for delivering user data

```
schemas:
    user:
      type: object
      additionalProperties: false
      properties:
        USER_ID:
          type: integer
          description: User ID
        USERNAME:
          type: string
          description: Login
        DISPLAY_NAME:
          type: string
          description: Full name
        STATUS:
          type: integer
          description: Status (0 - active, 1 - disabled, 2 - deleted)
          enum: [0, 1, 2]
        CREATE_DATE:
          type: string
          description: Create Date with format "Y-m-d H:i:s.u" in UTC
        EMAIL:
          type: string
          description: E-mail
        NOTE:
          type: string
```

```
          description: Note
```

Example:

Query

```
curl -X GET "https://server.company.com/xapi/user?start=1&limit=100" -H "accept: application/json" -H "X-Enable-OpenAPI: 1"
```

Response


```
{
  "data": [
    {
      "USER_ID": 0,
      "USERNAME": "string",
      "DISPLAY_NAME": "string",
      "STATUS": 0,
      "CREATE_DATE": "string",
      "EMAIL": "string",
      "NOTE": "string"
    }
  ],
  "meta": {
    "totalCount": 0,
    "start": 0,
    "limit": 0
  }
}
```


Getting Results of Completed Traffic Monitor Query

The chapter contains the following information:

- Receiving List of Queries;
- Starting Query Execution;
- Getting Status of Query;
- Getting Query Results.

Receiving List of Queries

Description: List of Traffic Monitor queries

Method: `GET`

Resource: `query`

Parameters (transferred as a JSON file):

- `userId` – id of the user, whose list of queries you need to receive;
- `start` – from which position to start sending queries (default value 0);
- `limit` – how many queries to send (default value 100, can not exceed 1000).

To your query you will get a list of existing TM queries with their ID and text name.

JSON scheme of the queries list

```
 "data": [
  {
    "QUERY_ID": 0,
    "DISPLAY_NAME": "string"
  }
],
"meta": {
  "totalCount": 0,
  "start": 0,
  "limit": 0
}
```

Example of query

```
curl -X GET "https://server.company.ru/xapi/query?userId=5&start=0&limit=10" -H "accept: application/json"
-H "X-Enable-OpenAPI: 1"
```

Starting Query Execution

Description: Starting query execution in Traffic Monitor

Method: `POST`

Resource: `query/{queryId}/exec`

Parameters:

- `queryId` – ID of a query you need to start.

JSON-scheme of response to query

```
{
  "data": {
    "QUERY_ID": 0,
    "QUERY_TYPE": "query",
    "DISPLAY_NAME": "string",
    "CREATE_DATE": "string",
    "CHANGE_DATE": "string"
  }
}
```

Example of query

```
curl -X POST "https://server.company.ru/xapi/query/0/exec" -H "accept: application/json" -H "X-Enable-
OpenAPI: 1"
```

Getting Status of Query

Description: Getting status of query

Method: `GET`

Resource: `query/{queryId}/status`

Parameters:

- `queryId` – id of the query, the status of which you need to receive.

JSON scheme of response

```
{
  "data": {
    "QUERY_ID": 0,
    "USER_ID": 0,
    "STATUS": 0,
    "RUN_DATE": "string",
    "COMPLETE_DATE": "string",
    "PERCENT_COMPLETE": 0,
    "OBJECT_COUNT": 0
  }
}
```

Example of query

```
curl -X GET "https://server.company.ru/xapi/query/0/status" -H "accept: application/json" -H "X-Enable-OpenAPI: 1"
```

Getting Query Results

Description: Traffic Monitor queries list of results

Method: `GET`

Resource: `query/{queryId}/result`

Parameters (transferred as a JSON file):

- `queryId` – id of query;
- `start` – from which position to start queries (default value 0);
- `limit` – how many queries to send (default value 100, can not exceed 1000);
- `fields:`
  - `OBJECT_ID` – event identifier in TM (positive integer);
  - `TBS_ID` – table space identifier (positive integer).

As a response to a query initiated earlier (see. *Starting Query Execution*) you will get a list of events with their ID.

JSON scheme of query

```
{
  "data": [
    {
      "OBJECT_ID": 0,
      "TBS_ID": 0
    }
  ],
  "meta": {
    "totalCount": 0,
    "start": 0,
    "limit": 0
  }
```

```
}
```

Example of query

```
curl -X GET "https://server.company.ru/xapi/query/5/result?start=0&limit=10" -H "accept: application/json"
-H "X-Enable-OpenAPI: 1"
```

12 Appendix A. Username and Password Creation Recommendations

Rules for logins

- The username should be 1-20 characters long.
- The username may include English letters, digits and underscore "_". It should start with a letter.

Rules for passwords

- The user password should be 8-128 characters long.
- The user password may include English letters, digits, and also the following characters:

«#», «$», «!» or «%».

- The password is case-sensitive.

Recommendations for strong passwords

- Recommended password length is 10-30 characters.
- The password must contain the combination of letters, digits and characters.
- It is not advised to:
    - include whole words or phrases into a password.
    - use a series of the same character.
    - start and end the password with the same character.
    - create a new password by adding characters to your current password

General recommendations

Do not start usernames or passwords with these combinations: SYS_ and ORA_.

Do not use Oracle reserved words in usernames or passwords:

ACCESS	DELETE	INTERSECT	OPTION	SUCCESSFUL
ADD	DESC	INTO	OR	SYNONYM
ALL	DISTINCT	IS	ORDER	SYSDATE
ALTER	DROP	LEVEL	PCTFREE	TABLE
AND	ELSE	LIKE	PRIOR	THEN
ANY	EXCLUSIVE	LOCK	PRIVILEGES	TO
AS	EXISTS	LONG	PUBLIC	TRIGGER
ASC	FILE	MAXEXTENTS	RAW	UID
AUDITBETWEEN	FLOAT	MINUS	RENAME	UNION
BY	FOR	MLSLABEL	RESOURCE	UNIQUE

CHAR	FROM	MODE	REVOKE	UPDATE
CHECK	GRANT	MODIFY	ROW	USER
CLUSTER	GROUP	NOAUDIT	ROWID	VALIDATE
COLUMN	HAVING	NOCOMPRESS	ROWNUM	VALUES
COMMENT	IDENTIFIED	NOT	ROWS	VARCHAR
COMPRESS	IMMEDIATE	NOWAIT	SELECT	VARCHAR2
CONNECT	IN	NULL	SESSION	VIEW
CREATE	INCREMENT	NUMBER	SET	WHENEVER
CURRENT	INDEX	OF	SHARE	WHERE
DATE	INITIAL	OFFLINE	SIZE	WITH
DECIMAL	INSERT	ON	SMALLINT	
DEFAULT	INTEGER	ONLINE	START	

13 Appendix B. Monitoring Indicators

The table below is listing all indicators, used by monitoring subsystem to check the state of System components, installed and operating on servers in All-in-one mode. The check includes obtaining indicator values at regular intervals and comparing them with threshold values. Critical values of all indicators may mean that the server is physically inaccessible. In this case, please check if the server works properly.

Warning!

If configuration files have been changed, then the threshold values and check period may differ from the specified values.

Warning!

If recommendations from the table below do not help you to solve the problem, please contact InfoWatch technical support at *support@infowatch.com*.

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
General System load	Load of IW-Linux-servers, TMcap servers, DB servers over the last 15, 5, and 1 minute	30 minutes	🔴 – at least one figure exceeds the value of 20.0, 16.0 and 12.0 for 15, 5 and 1 minute respectively 🟡 – at least one figure exceeds the value of 10.0, 8.0 and 6.0 for 15, 5, and 1 minute respectively 🟢 – none of the figures exceed the value of 10.0, 8.0 and 6.0 for 15, 5 and 1 minute respectively	Check the load of interception and analysis servers, and DB servers: 1. Run `top` 2. Sort the results, using the commands: a. Shift+m to sort by RAM load; b. Shift+p to sort by CPU load.
System components statuses	System services statuses	30 minutes	🔴 – at least one service is not started 🟢 – all services are started	In All-in-one installation mode, the list and number of System services are known and permanent. To display services status

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
				details, type `iwtm status`
Syslog service status	Whether the `syslog` service is started or not	30 minutes	🔴 – service not started 🟢 – service started	Check if `rsyslog` service is running: `systemctl status rsyslog`
Number of active users	Number of users (Security Ofiicers), registered with the System (Linux servers)	30 minutes	🔴 – > 50 users 🟡 – from 20 to 50 🟢 – < 20 users	It is necessary to decrease number of security officers to 50 persons
DB Alert Log Errors	Whether Alert Log contains errors (DB servers)	5 minutes	🔴 – errors found 🟢 – no errors	It is necessary to decrease the number of errors having `SEVERITY=HIGH` label. **Search in Alert Log should be based on the following words:** • ORA- • Failed to archive • ORACLE Instance .* - Archival Error • ARC[0-9]: I/O error • ARC[0-9]: RFS network connection lost at host • ARC[0-9]: Error 3113 creating • Media Recovery failed with error • Corrupt block • Fractured block • Bad header • Data in bad block • Reread of rdba • log corruption • RESETLOGS **Search in pg_log should be based on the following words:** • FATAL • PANIC

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
				• ERROR
Server availability	Response time for sent packages and number of lost packages (Linux servers)	5 minutes	🔴 – > 500 ms 🟡 – from 100 to 500 ms 🟢 – < 100 ms	Ping results with the following parameters `100.0,20%! 500.0,60%` show that: Response to sent packages exceeds 100 milliseconds and more than 20% of packages are lost; Response to sent packages exceeds 500 milliseconds and 60% of packages are lost.
Availability of incorporated mail transfer agent	Postfix server availability	30 minutes	🔴 – not available 🟢 – available	1. Check Postfix server status: s `ystemctl status postfix` 2. Start checking correctness of configuration file `/etc/postfix/main.cf`, using the command `postfix check`
Server availability via SSH	Response time	10 minutes	🔴 – >= 10 seconds 🟢 – < 10 seconds	Check `sshd` daemon status, using the command `systemctl status sshd`
Device Monitor server availability	Response for request	30 minutes	🔴 – no connection 🟢 – connection established	If the System has an active Device Monitor license, it should be displayed. Check the license in **Administration** -> **Licenses** in Traffic Monitor Management Console.
Using swap file	Free space in swap (virtual memory) (Linux servers)	5 minutes	🔴 – < 10% free 🟡 – from 10 % to 50% free 🟢 – > 50% free	For details, see *InfoWatch Traffic Monitor. Installation Guide. Recommendations on Disk Space Partitioning*

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
Free space on root partition	Free space in the root file system (Linux servers)	30 minutes	🔴 – < 10 000 MB 🟡 – from 10 000 MB to 20 000 MB 🟢 – > 20 000 MB	
Free space on `/var` partition	Free space in the `/var.` file system `(` Linux servers)	30 minutes	🔴 – < 10 000 MB 🟡 – from 10 000 MB to 20 000 MB 🟢 – > 20 000 MB	
Status of time synchronization service	`ntpd.` service status (NTPD)	30 minutes	🔴 – NTP server is unavailable 🟢 – NTP server is available	Check `ntpd` daemon status, using the command `systemctl status ntpd`
System time offset	Time lag on System servers and NTP server (Linux servers)	360 minutes	🔴 – > 40 seconds 🟡 – from 20 to 40 seconds 🟢 – < 20 seconds	The error occurs, if the System cannot connect NTP servers, or System configuration has no NTP servers. In this case, the error message "*Can't create socket connection*" will appear. Make sure that `ntp.conf` and `iwmon-services-ntp.cfg` configuration files have the same correct NTP server specified.
Core dump	Checking the functionality of services, the absense of failures, the number of files with in the `/opt/iw/tm5` directory that look like: • core_<....>	30 minutes	🔴 – 5 or more core dumps 🟡 – 1 to 4 core dumps 🟢 – no core dumps	Contact InfoWatch technical support at *support@infowatch.com*

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
	• <...>.core (Linux-servers)			

Queue of events processing errors	Number of objects in **queue/final-errors** queue (`iw_rammer`)	1 minute	🔴 – > 1 objects
			🟢 – 0
				Option 1: Clear queues using the commands:
				`su - iwtm`
				`/opt/iw/tm5/bin/iw_qtool erase /opt/iw/tm5/queue/final-errors`
				`exit`
				Option 2:
				1. Open the `/etc/cron.d/iwtm_cleanup` file in the edit mode.
				2. Reduce the period of cleaning files in file queue folders. You can do it in a regular `cron` job (the default value is 30 days).
				3. Save changes and close the file.
				For more details see Inline Mode

Database status	Database status (depending on the DBMS in use)	10 minutes	🔴 – <> ok 🟢 – ok	Check availability of the database: 1. Log in to the database: `s qlplus iwtm@iwtm` 2. Enter password `xxXX1234` 3. Watch `SQL>`
Database access service status	Oracle TNS listener's status	10 minutes	🔴 – TNS listener is not started	Indicator is enabled and displayed in the list only in case when Oracle SMDB is used in the System

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
			🟢 – TNS listener is started	
The number of Oracle data files	If the number of files has reached 90% of the specified maximum value	60 minutes	🔴 – if the current number of data files reaches the maximum value 🟡 – if the number of files is 90% of the maximum value 🟢 – if the number of files is fewer than 90% of the maximum value	If the number of files reaches the maximum value, you should increase the value of the `db_files` Oracle paramerer
Free space on the database main partition	Free space on the partition that stores main information (By default: `/u01`)	30 minutes	🔴 – < 10 000 MB 🟡 – from 10000 to 20 000 MB 🟢 – > 20 000 MB	Dedicate more disk space for `/u01` (for details, see *InfoWatch Traffic Monitor. Installation Guide. Recommendations on Disk Space Partitioning*)
Free space in database events catalogue	Free space on the partitions that store events (By default: `/u02`)	30 minutes	🔴 – < 7% 🟡 – from 7 to 15% (if daily table spaces are stored in several sections, then when the value is < 7% in one section, at least) 🟢 – > 15%	If daily table spaces are stored in several sections, then the free space is calculated as a total sum of all sections, which during the installation were dedicated for storing daily table spaces. For details, see *InfoWatch Traffic Monitor. Installation Guide. Recommendations on Disk Space Partitioning*
Free space on daily events partition on fast disk	Free space on fast disk on partition that stores daily table spaces	30 minutes	🔴 – < 7% 🟡 – from 7 to 15% 🟢 – > 15%	The indicator is added, if during the installation *Fast and slow disks* storing mode was chosen.

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
				For details, see *InfoWatch Traffic Monitor. Installation Guide. Recommendations on Disk Space Partitioning*
Database patch availability	Whether Oracle patches, required for correct operation of the System, are installed	Once a day	🔴 – not installed 🟢 – installed	Indicator is on and displayed in the list only if Oracle DBMS is used in the System. Download and install the required patch.
Size of database alert log	Size of PostgreSQL DBMS alert log: System LogSize of Oracle DBMS alert log: Alert Log	5 minutes	🔴 – > 3 GB 🟡 – from 1 GB to 3 GB 🟢 – < 1 GB	Contact InfoWatch technical support at *support@infowatch.com*
DB syslog errors	Database log contains errors	30 minutes	🔴 – > 1 errors found 🟡 – 1 error found 🟢 – no errors found	It is necessary to decrease the number of errors with `SEVERITY=Error`
Mail traffic processing queue	Number of objects in queue **queue/smtp** (`iw_messed`)	30 minutes	🔴 – > 10000 objects 🟡 – from 1000 to 10000 objects 🟢 – < 1000 objects	1. Check status of service: `s ystemctl status iw_messed` 2. View service logs in `opt/iw/tm5/log/messed_app.log` 3. Check RAM and swap load: a. For a short response, run command `free -m` b. For a detailed response, run command `cat /proc/meminfo`

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
Queue for specific analysis for xAPI/pushAPI objects	Number of objects in queue **queue/analysis** (`iw_analysis`)	30 minutes	🔴 – > 10000 objects 🟡 – from 1000 to 10000 objects 🟢 – < 1000 objects	1. Check status of service: `s systemctl status iw_analysis` 2. View service logs in `opt/iw/tm5/log/analysis.log` 3. Check RAM and swap load: a. For a short response, run command `free -m` b. For a detailed response, run command `cat /proc/meminfo`
Queue of objects loading from interceptors to **iw_x2x** service	Number of objects from interceptors for **iw_x2x** queueing in **queue/db** (`iw_x2x`)	30 minutes	🔴 – > 10000 objects 🟡 – from 1000 to 10000 objects 🟢 – < 1000 objects	1. Check status of service: `s systemctl status iw_x2x` 2. View service logs in `opt/iw/tm5/log/x2x.log` 3. Check RAM and swap load: a. For a short response, run command `free -m` b. For a detailed response, run command `cat /proc/meminfo`
Loading to storage queue	Number of objects queueing in **queue/x2x**, which are to be loaded to storage by service **iw_x2db** (TMcap servers)	30 minutes	🔴 – > 10000 objects 🟡 – from 1000 to 10000 objects 🟢 – < 1000 objects	1. Check status of service: `s systemctl status iw_x2db` 2. View service logs in `opt/iw/tm5/log/x2db.log`

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
				3. Check RAM and swap load: a. For a short response, run command `free -m` b. For a detailed response, run command `cat /proc/meminfo`
Queue of applied policies actions processing	Number of objects queueing in **queue/ blackboard** (Linux servers)	30 minutes	🔴 – > 1000 objects 🟡 – from 200 to 1000 objects 🟢 – < 200 objects	1. Check status of service: `systemctl status iw_analysis` 2. View service logs in `opt/iw/tm5/log/analysis.log` 3. Check RAM and swap load: a. For a short response, run command `free -m` b. For a detailed response, run command `cat /proc/meminfo`
Queue of objects to be indexed in Sphinx for full-text search	Number of objects queueing to be indexed in Sphinx, for full-text search	30 minutes	🔴 – > 10000 objects 🟡 – from 1000 to 10000 objects 🟢 – < 1000 objects	1. Check status of service: `searchd --status -c /etc/infowatch/sphinx.conf` 2. View service logs in `opt/iw/tm5/log/sphinx/` 3. Check RAM and swap load: a. For a short response, run command `free -m` b. For a detailed response, run command

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
				`cat /proc/meminfo` 4. Calculate the number of running `searchd` instances: a. If `iw_is` is running on a node, then: 1 (`iw_is itself)` + 4 (`grep max_children /opt/iw/tm5/etc/is.conf)` = 5 b. If `iw_indexer` is running on a node, then: 1 (`iw_ndexer itself)` + 8 (`grep max_children /opt/iw/tm5/etc/sphinx.conf)` = 9 c. Total: a+b=14 `searchd` instances. The more processes, the less are the odds of slow user request processing.
Queue of objects to be indexed in Sphinx for meta-information search	Number of objects queueing to be indexed in Sphinx, for meta-information search	30 minutes	🔴 – > 10000 objects 🟡 – from 1000 to 10000 objects 🟢 – < 1000 objects	1. Check status of service: `searchd --status -c /etc/infowatch/sphinx.conf` 2. View service logs in `opt/iw/tm5/log/sphinx/` 3. Check RAM and swap load: a. For a short response, run command `free -m` b. For a detailed response, run command

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
				`cat /proc/meminfo`
Events indexation by **iw_indexer** service number of errors	Indexation errors number (`iw_indexer`)	30 minutes	🔴 – > 100 errors 🟡 – from 10 to 100 errors 🟢 – 10 errors	1. Check status of service: `s ystemctl status iw_indexer` 2. View service logs in `opt/iw/tm5/log/indexer.log` 3. Check RAM and swap load: a. For a short response, run command `free -m` b. For a detailed response, run command `cat /proc/meminfo`
Number of meta-information indexation by **iw_is** service errors	Number of errors occurred during indexation of meta-information by **iw_is** service	30 minutes	🔴 – > 100 errors 🟡 – from 10 to 100 errors 🟢 – 10 errors	1. Check status of service: `s ystemctl status iw_is` 2. View service logs in `opt/iw/tm5/log/is.log` 3. Check RAM and swap load: a. For a short response, run command `free -m` b. For a detailed response, run command `cat /proc/meminfo`
iw_is commands queue	Number of commands in **iw_is** service queue	30 minutes	🔴 – > 100 commands 🟡 – from 50 to 100 commands 🟢 – < 50 commands	
Data selection queues	Number of objects in **iw_is** service queue	30 minutes	🔴 – > 100 objects 🟡 – from 50 to 100 objects 🟢 – < 50 objects	
iw_is indexation queue	Number of commands in **iw_is** service queue	30 minutes	🔴 – > 100 objects 🟡 – from 50 to 100 objects	

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
			🟢 – < 50 objects	
iw_is errors queue	Whether **iw_is** service has errors	5 minutes	🔴 – > 500 errors found 🟡 – from 50 to 500 errors found 🟢 – < 50 errors found	
Queue of commands for text tablespace	Number of commands in text tablespace queue	30 minutes	🔴 – > 50 commands 🟡 – from 20 to 50 commands 🟢 – < 20 commands	1. Enable logging of **iw_indexer** service work on `debug` (see *Logging System Work*). 2. Check if log file `iw_indexer` in `opt/iw/tm5/log/indexer.log` has errors. 3. Contact InfoWatch technical support.
Queue of commands for meta-information tablespace	Number of commands in meta-information tablespace queue	30 minutes	🔴 – > 50 commands 🟡 – from 20 to 50 commands 🟢 – < 20 commands	

Events processing errors queue	Number of objects in errors queue **queue/errors** (Extractors Framework, TMcap servers: `iw_luaengined`, `iw_warpd`, `iw_cas`)	30 minutes	🔴 – > 500 objects
			🟡 – from 50 to 500 objects
			🟢 – < 50 objects
				Option 1: Clear queues by using the commands:
				`su - iwtm`
				`/opt/iw/tm5/bin/iw_qtool erase /opt/iw/tm5/queue/errors`
				`exit`
				Option 2:
				1. Open the `/etc/cron.d/iwtm_cleanup` file in the edit mode.
				2. Reduce the period of cleaning files in file queue folders. You can do it in a regular `cron` job (the default value is 30 days).
				3. Save changes and close the file.

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations

Errors queue of processing events by **iw_x2x**	Number of objects in queue **queue/x2x-errors** (`iw_x2x`)	30 minutes	🔴 – > 500 objects
			🟡 – from 50 to 500 objects
			🟢 – 50 objects
				Option 1: Clear queues by using the commands:
				`su - iwtm`
				`/opt/iw/tm5/bin/iw_qtool erase /opt/iw/tm5/queue/x2x-errors`
				`exit`
				Option 2:
				1. Open the `/etc/cron.d/iwtm_cleanup` file in the edit mode.
				2. Reduce the period of cleaning files in file queue folders. You can do it in a regular `cron` job (the default value is 30 days).
				3. Save changes and close the file.

Errors queue of processing events by **iw_x2db** service	Number of objects in queue **queue/x2db-errors** (`iw_x2db`)	30 minutes	🔴 – > 500 objects
			🟡 – from 50 to 500 objects
			🟢 – 50 objects
				Option 1: Clear queues by using the commands:
				`su - iwtm`
				`/opt/iw/tm5/bin/iw_qtool erase /opt/iw/tm5/queue/x2db-errors`
				`exit`
				Option 2:
				1. Open the `/etc/cron.d/iwtm_cleanup` file in the edit mode.
				2. Reduce the period of cleaning files in file queue folders. You can do it in a regular `cron` job (the default value is 30 days).

Indicator name	What is checked, where it is checked	Poll interval	Values	Details and recommendations
				3. Save changes and close the file.

Errors queue of applied policies actions processing	Number of objects in queue **queue/blackboard_errors** (Linux servers)	30 minutes	🔴 – > 500 objects
			🟡 – from 50 to 500 objects
			🟢 – 50 objects
				Option 1: Clear queues by using the commands:
				`su - iwtm`
				`/opt/iw/tm5/bin/iw_qtool erase /opt/iw/tm5/queue/blackboard-errors`
				`exit`
				Option 2:
				1. Open the `/etc/cron.d/iwtm_cleanup` file in the edit mode.
				2. Reduce the period of cleaning files in file queue folders. You can do it in a regular `cron` job (the default value is 30 days).
				3. Save changes and close the file.