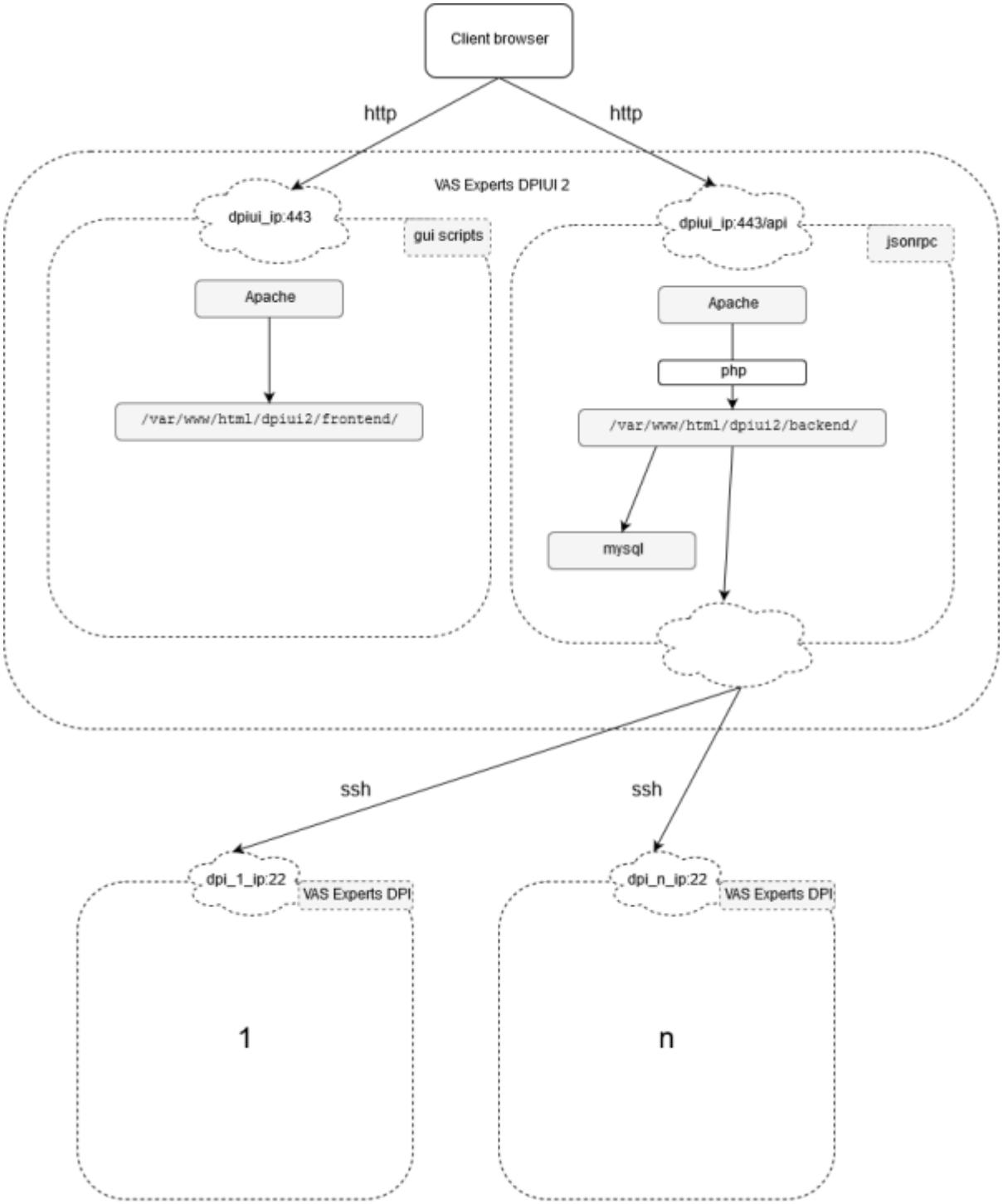
Содержание

The VAS Experts DPI Graphical User Management Interface ver.2

Introduction

The VAS Experts DPI user Management Interface is designed to control the DPI using the graphical user interface.

Architecture



Installation and upgrading

Equipment prerequisites

It is desired to use equipment or virtual machines within the subsystem with the following characteristics:

1. Central Processor Unit (CPU) 2.5 GHz, 1 pce
2. Random access memory (RAM) 512 MB - 1 GB

3. Hard disk drive (HDD) 50 GB - 250 GB
4. Operating System Cent OS 6.4 - 7.3
5. Network interface card (NIC) 10 Mbps and above

Recommended operating system is Cent OS 7+. Make sure that supervisor 3+ is installed if you need to install the VAS Experts DPI user Management Interface on Cent OS 6. If you do not have the package, please contact our technical support.

Do not install the subsystem on the same hardware with the DPI! Use a separate virtual machine instead.

Before the installation

New virtual machine

1. Make sure that openssh-clients is installed - it is required in order to connect to the DPI
2. All other necessary environments will be installed automatically

Existing virtual machine

1. Make sure that the openssh-clients is installed - it is required to connect to DPI
2. If PHP version < 7.1 is installed, uninstall the old version:

   ```
   yum -y remove php*
   ```

   The new one will be automatically installed during the dpiui2 installation.

3. If MySql is installed, please, delete:

   ```
   yum remove mysql mysql-server mysql-community-common
   ```

   Also MySql directory has to be purged:

   ```
   mv /var/lib/mysql /var/lib/mysql_old_backup
   ```

   When installing the dpiui2 MariaDB 10.2+ will be installed

CentOS 6

Recommended operating system is Cent OS 7+. Make sure that supervisor 3+ is installed if you need to install the VAS Experts DPI user Management Interface on Cent OS 6. If you do not have the package, please install it issuing the following commands:

```
sudo wget https://vasexperts.ru/install/supervisor-3.0-1.gf.el6.noarch.rpm
yum install supervisor-3.0-1.gf.el6.noarch.rpm
```

Installation

Watch a tutorial (english subs): https://www.youtube.com/watch?v=-Nzh0jb2fyM&feature=emb_logo

Before installing or upgrading, check your Internet connection. Make sure you run scripts as root or run it using sudo.

Attention!: You have to disable selinux. To do that, set

```
SELINUX = disabled
```

in the `/etc/selinux/config` file and restart the server.

To install or upgrade, run the script: dpiui2-rpm_install.sh

```
sudo wget https://vasexperts.ru/install/dpiui2-rpm_install.sh
sudo sh dpiui2-rpm_install.sh
```

which installs the dpiui2 rpm package. All the needed settings will be done automatically according to your system current configuration.

The installation process will install/update the following environment:

1. PHP >= 7.1
2. MariaDB >= 10.2
3. Apache
4. Composer
5. PHP SSH2 lib
6. Laravel/Lumen

Required ports will be opened, as well as cron will be launched to perform background tasks on a schedule.

The subsystem will be installed to the

```
/var/www/html/dpiui2/
```

directory

After installation, type in your browser address bar:

```
https://<VM_IP_address>/
```

Attention!:https is used (not http)

The following account will be created by default:

1. Login - admin
2. Password - vasexperts

Upgrading

To upgrade the previously installed version, run the script from the Installation section.

Do not forget to close/open the browser after upgrade. It is also desirable to perform relogin.

Configuration

Watch a tutorial (english subs): https://www.youtube.com/watch?v=KHqYpV3HXqU&feature=emb_logo

Subsystem configuration is done by editing the .env file

```
/var/www/html/dpiui2/backed/.env
```

The file has the following content:

```
#System settings. It should remain unchanged.
APP_ENV=local
APP_DEBUG=true
APP_KEY=
APP_TIMEZONE=UTC

#Application URL. It is needed to form the correct link when sending QoE
reports to e-mail
APP_URL=https://localhost/

#System settings regarding connection to the MySql DB, it should remain
unchanged
DB_CONNECTION=mysql
DB_HOST=localhost
DB_PORT=3306
DB_DATABASE=dpiui2
DB_USERNAME=root
DB_PASSWORD=vasexperts
```

```
#Settings regarding connection to the SMTP server. They are needed to send
email notifications.
CFG_SMTP_UNAME=dpiuitest@gmail.com
CFG_SMTP_PW=dpiuitestdpiuitest
CFG_SMTP_HOST=smtp.gmail.com
CFG_SMTP_PORT=587
#tls or ssl
CFG_SMTP_SECURE=tls

#Technical support address
CFG_SEND_ERROR_EMAIL=sd@vas.expert
#Address for sending of email copies
CFG_SEND_COPY_EMAIL=

#System settings, prohibited from changing
CACHE_DRIVER=file
QUEUE_DRIVER=database
SESSION_DRIVER=cookie

#Settings regarding connection to QoE Stor
QOESTOR_DB_HOST=localhost
QOESTOR_DB_PORT=8123
QOESTOR_DB_USER=default
QOESTOR_DB_PASS=''
QOESTOR_DB_NAME=qoestor
QOESTOR_CACHE_LIFE_TIME_SEC=3600
QOESTOR_MAIN_LOG_PARTITIONS_LIFE_TIME_HOUR=24
QOESTOR_AGG_LOG_PARTITIONS_LIFE_TIME_DAYS=15

#Subscriber synchronization period in minutes (for the Subscribers and
Services and Advertising sections)
SM_SUBSCRIBERS_UPDATE_PERIOD_MINUTES=30

#Data cleanup period for charts in the Performance Section
CHART_DATA_DELETE_DAYS_INTERVAL=60

#CG-NAT profile and statistics synchronization period
CG_NAT_SYNC_MINUTES_INTERVAL=5

#Хост Vas Cloud
VAS_CLOUD_HOST=5.200.37.122
```

If changes to .env have been made, you should run the following command:

```
php /var/www/html/dpiui2/backend/artisan queue:restart
```

DPI connection details

Whatch a tutorial (english subs): https://www.youtube.com/watch?v=81WMPGw6tak&feature=emb_logo

New user should be created with `sudo` access granted on the DPI device.

Let's consider `dpisu` user creating as an example:

1. Create the `dpisu` user

```
adduser dpisu
passwd dpisu
```

2. Add to the /etc/sudoers.d/dpisu file the following stuff

```
Defaults:dpisu !requiretty
Defaults secure_path =
/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin
dpisu    ALL=(ALL)       NOPASSWD: ALL
```

3. Grant write permissions to the **directories**

```
chmod 777 /tmp/dpi
chmod 777 /etc/dpi/.save
```

Setting up a connection to the QoE Stor

This section appeared in version 2.1.0.

For QoE analysis, you should to install and configure the QoE Stor module.

To connect to the QoE Stor module database, you should add following settings (listed below) to the .env file (see Configuration)

```
QOESTOR_DB_HOST=localhost
QOESTOR_DB_PORT=8123
QOESTOR_DB_USER=default
QOESTOR_DB_PASS=''
QOESTOR_DB_NAME=qoestor
QOESTOR_CACHE_LIFE_TIME_SEC=3600
QOESTOR_MAIN_LOG_PARTITIONS_LIFE_TIME_HOUR=2
QOESTOR_AGG_LOG_PARTITIONS_LIFE_TIME_DAYS=14
```

Here

- QOESTOR_DB_NAME - database name, always equals to 'qoestor'
- QOESTOR_DB_HOST - hostname or address where the QoE Stor module is installed

- QOESTOR_DB_PORT - port, by default it equals to 8123
- QOESTOR_DB_USER - DB username, by default it equals to 'default'
- QOESTOR_DB_PASS - password is empty by default
- QOESTOR_CACHE_LIFE_TIME_SEC - user reports cache storage period in seconds, by default equals to 3600 sec. (1 hour)
- QOESTOR_MAIN_LOG_PARTITIONS_LIFE_TIME_HOUR - the storage period of the main netflow and clickstream logs in hours, by default equals to 2 hours
- QOESTOR_AGG_LOG_PARTITIONS_LIFE_TIME_DAYS - the storage period of aggregated netflow and clickstream logs in days, by default equals to 14 days

If changes to .env have been made, you should run the following command:

```
php /var/www/html/dpiui2/backend/artisan queue:restart
```

Version Information

Version v.2.10.14 (04/09/2020)

- Updated protocol dictionaries
- Added push notifications:

1. About connecting to a device
2. Data synchronization and
3. Notifications from the section "Triggers and Notifications"

- Added section for viewing and downloading DPIUI2 logs
- Added QoE Store configuration section via DPIUI2
- Added section for viewing and downloading QoE Store logs
- Fixed interface bugs in existing sections

Version v.2.9.5 (12/25/2019)

- Congratulations on NG 2020 and the opportunity to receive a gift!
- New section DPIUI2 server configuration
- New DPIUI2 update section: the ability to update and auto-update the dpiui2 version
- Tariffs refactoring
- Equipment section refactoring
- In the JSON API of the QoE section, the parameter for the maximum number of entries is made in the settings
- Minor bug fix

Version v.2.8.2 (06.11.2019)

- SSH bruteforce report
- System trigger with bruteforce SSH report
- Filtering by CIDR has been fixed

- Personal account for operator's subscribers
- Fixes in the Hotspot section
- Usability fixes: text filter in tables, service profile selection in the subscriber’s card, codes in services are replaced by names
- Bug concerning removal of Mini Firewall and CGNAT services from a subscriber has been fixed
- Check for class sum when using the static keyword

Version v.2.7.9 (25.10.2019)

- VAS Cloud access using domain name
- Locked in the clickstream
- Critical bug in synchronization of subscribers service profiles has been fixed

Version v.2.7.7 (16.09.2019)

- Added new Mini firewall section
- Raw log analytics: Protocol and subscriber report sets
- VAS Cloud personal account: new section Charges and payments, new section Statistics, a new feature to request withdrawal of funds by the operator
- Incorporation of logs Clickstream and Netflow (the both raw and aggregated). Analytical reports on such logs.

Version v.2.6.6 (16.09.2019)

- A new HotSpot management section is added
- Further improvements in the QoE/Logs section: speaker filters, column filters, Reset Filters button

Version v.2.5.7 (06.09.2019)

- Adaptation to the new version of the QoE Store (fastor-1.1.1)
- Optimization of jobs for cleaning outdated partitions in QoE Stor
- Added "Directory of excluded subnets" and "Directory of excluded AC numbers" dictionaries in QoE/Administrator

Version v.2.5.2 (16.08.2019)

- Added a new section Triggers and notification
- Added licensing via VAS Cloud
- Bugfix in the QoE Analytics and CG-NAT sections

Version v.2.4.1 (11.07.2019)

- Issue with the absence of subscribers list in the Black and White Lists section has been fixed
- Refactoring of dpiui2 and fastdpi update check functions

- Fixed an issue concerning occasionally hanging of ssh sessions

Version v.2.4.0 (08.07.2019)

A new CG-NAT section was added

- Added the feature to manage the CG NAT service (service 11)
- Added the feature to monitor the load level of CG NAT pools
- Support for all the innovations in the CG NAT service added in the VAS Experts DPI 8.3.1

Miscellaneous

- Added filter "Number of sessions" in the QoE section / Subscribers
- Added the feature to save the filter in all QoE sections
- ДBackground check for a new version of fastdpi has been added

Version v.2.3.4 (27.05.2019)

- Fixed a number of issues in the Black and White Lists section
- Added “Use federal” option
- Added the reature to see the name of the service profile installed on DPI in the Tariffs, Black and White Lists sections

Version v.2.3.1 (21.05.2019)

Added a new section Services / Advertising & Ad blocking

Added the "Vas Cloud Services" new section / Personal account and Vas Cloud / Vas ADS

In the Services / Subscribers and services

- Added the feature to apply a profile to the Black and white lists services
- Added the feature to apply a tariff plan

In the Services / Black and White Lists section

- Redesigned the mechanism for binding subscribers to the service
- Improved sync
- Improved import of lists from QoE
- Fixed a number of bugs

In the Services / Tariffs

- Improved synchronization of individual tariffs

In the Performance section

- Added an option to clear data for charts - CHART_DATA_DELETE_DAYS_INTERVAL parameter in .env
- Fixed incorrect displaying of a large number of CPUs on the CPU usage diagram

- Fixed incorrect displaying of clusters on traffic graphs on DNA interfaces

In the QoE Analytics section

- Added the feature to filter by IP addresses based on the mask (CIDR)
- Fixed several bugs

Miscellaneous

- A warning is displayed if fastdpi is not running
- Added the feature to automatically check for a new version of DPIUI2
- Added the feature to view the license on the DPI device

Version v.2.2.0 (25.03.2019)

A new Tariff plans section is added

- Tariff Management
- Enabling/disabling of tariff plan to Subscribers
- Tariff plan import option

In the Analytics section

- A bug concerning the incorrect counting the subscribers number in the QoE Dashboard section is fixed
- New feature to set conditions for filtering traffic
- Report on Clickhouse processes. The ability to stop the process
- Report with charts on AS traffic
- Legends in series charts are shown in the tabular form.
- Traffic volumes reports
- New feature to see the active report status is added.
- Ability to stop report building is added.
- Retransmit Reports

In Configuration and Protocol prioritization (DSCP) sections

- Policing form is improved
- A bug with displaying protocol names containing spaces is fixed

In Subscribers and services section

- An issue regarding adding a subscriber without binding to the login is fixed
- New options are added to the filters: "No services", "No tariffs"

Version v.2.1.14 (21.02.2019)

Regarding "QOE ANALYTICS" section

- Query history widget (filters) is added
- Quick period of time selection widget is added
- Reports on Traffic are added

- “Mini” reports (with a minimum number of columns to speed up data processing) are added
- Feature to save the state of sections, jumping to desired tabs by link
- Columns sorting in reports (by default descending order is used) is added
- Administrator section (the number of options will be expanded) is added
- Many bugs have been fixed

Regarding "DPI CONTROL/PERFORMANCE" section

- Charts of loading of cores of the processor (online and statistics) depending on time are added
- "Traffic on the interface" charts (Kbit/s) are added
- Some bugs have been fixed

Regarding "DPI CONTROL/CONFIGURATION" section

- A new feature to enable/disable displaying of comments to parameters is added
- Cosmetic tweaks

Regarding "SERVICES CONTROL" section

- New subsection "Black and white lists" under the "Services control/Services" is added

Regarding "SERVICE MANAGEMENT/ADVERTISING" section

- Import of categories for black and white lists from QoE
- Synchronization issues are fixed
- Error message displaying is added in case the subscriber for any reason is not added to the advertising campaign.

Version v.2.1.11 (21.12.2018)

- Bugs regarding filters in QoE sections are fixed
- Bugs in the Prioritization by protocols (dscp) section are fixed
- Bugs in the Prioritization section for autonomous systems (AS) are fixed
- Bugs in the Subscribers and Services section are fixed. FTTB logins and IPv6 support is added.
- Bugs in ADVERTISING section are fixed.
- New feature to force synchronization of subscribers and campaigns appeared. Update period setting is added.

Version v.2.1.10 (03.12.2018)

- A bug manifesting when user creates a cache, resulting in write operation interruption (under certain circumstances)
- Reports and filters by host categories are added

Version v.2.1.9 (22.11.2018)

- Export to Excel and other formats
- Export to PNG for charts
- Feature to select the type of charts

- Reports on raw click stream and flow logs using filters and features to export
- Clearing aggregated logs

Version v.2.1.7 (13.11.2018)

- A bug regarding filtering by time period is fixed. Now only the necessary partitions are queried, not the entire table.
- Cache has been optimized. Now only the totals are stored in the cache, rather than intermediate data.
- Bugs in "RTT distribution" and "Top subscribers with high RTT" reports are fixed

Version v.2.1.5 (07.11.2018)

- Bug related to unresponsive hung SSH sessions is fixed in version 2.1.4.
- Other hot-fixes.

This version works with the QoE Stor 1.0.4+ module

Version v.2.1.4 (02.11.2018)

The changes mainly concern the “QoE ANALYTICS” section and the QoE Stor module.

- Memory consumption optimization (QoE Stor module) when generating reports is implemented
- Background reports generating

This version works with the QoE Stor 1.0.4+ module

Version v.2.1.0 (20.09.2018)

- New QoE ANALYTICS section is added
- New ADV CONTROL section is added

Versions v.2.0.0 - v.2.0.6

- Further improvements
- Bug fixes

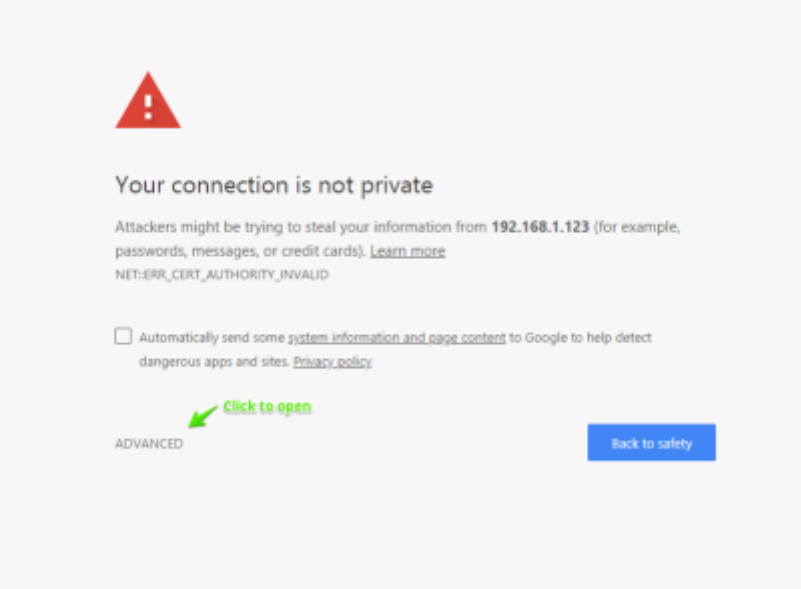
Version v.2.0.0 (28.12.2017)

- Interface is significantly improved
- Usability is significantly improved

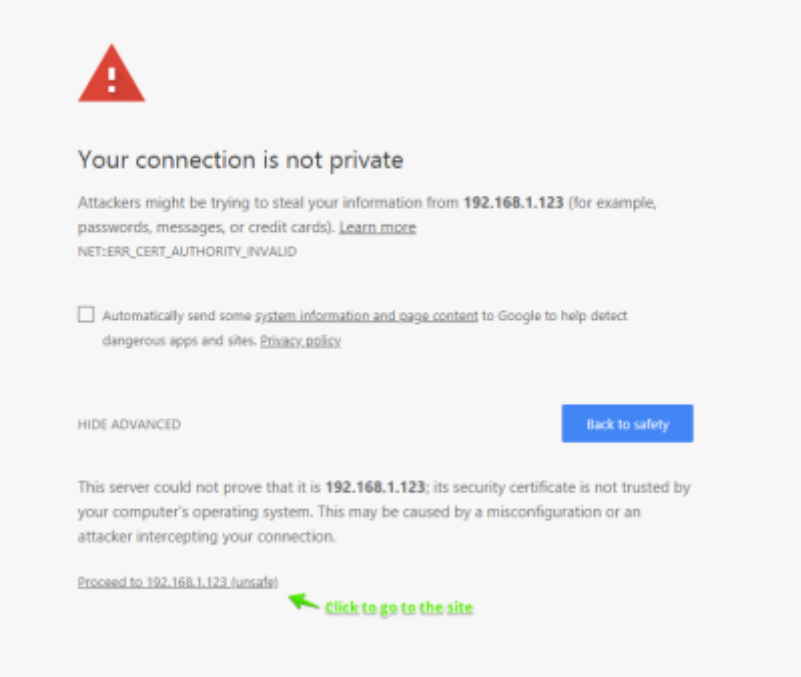
User Guide

Authorization and first launch

Since a self-signed ssl certificate is installed, the window that follows will appear in the browser (in different browsers it appears in different ways, in the screenshot below Google Crome is shown):

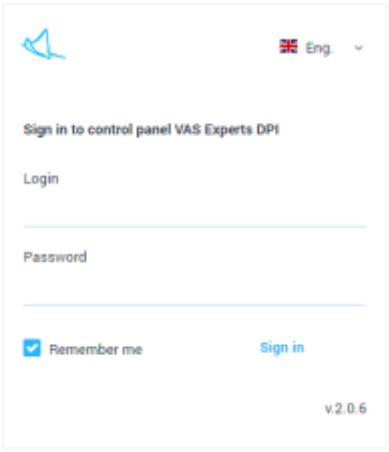


Click on "ADVANCED" to open. The window will appear:



Press "Proceed to ..." to the site.

At the first start the login window opens. You need to supply a username and a password.



If the "Remember me" check box is selected the session is remembered and the authorization is no longer required.

During the installation the following user is created by default:

1. Login - admin
2. Password - vasexperts

If you are logged in for the first time, you need to configure the hardware. See the section Managing the list of devices

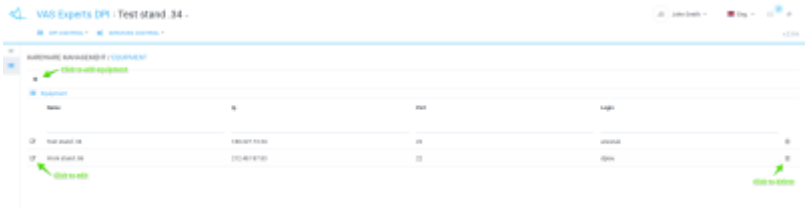
Device Management

Section contents:

Managing the list of devices

To switch to the section "Managing the list of devices" open the "Device_name" menu and click "Managing the list of devices".

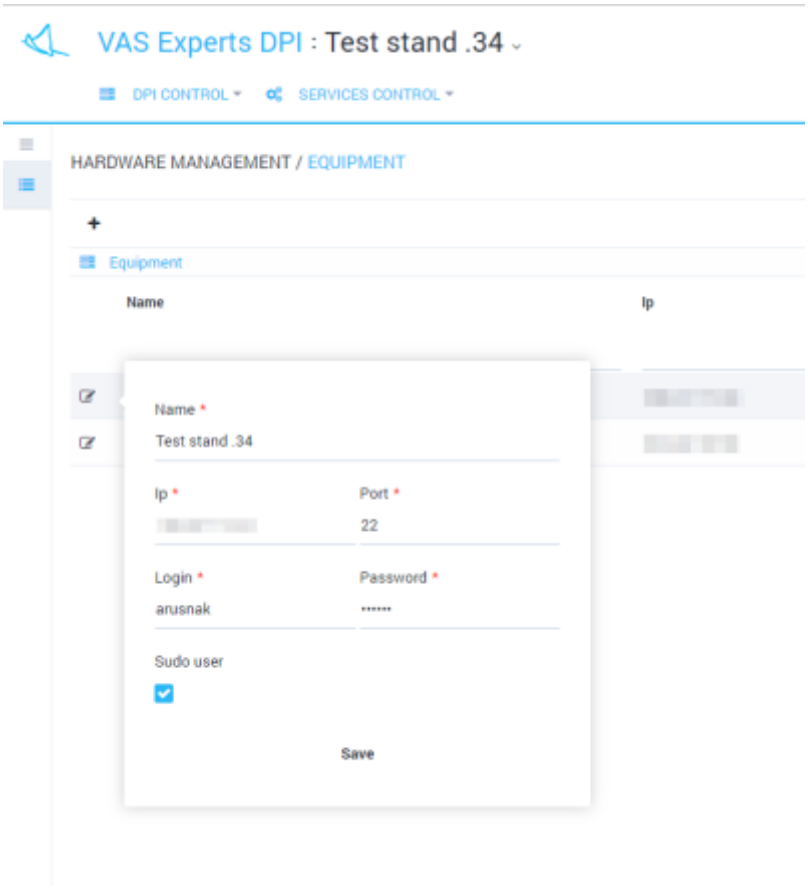
The section looks like the figure below.



This section implements the following features:

- View a list of devices along with filtering feature
- Add new device
- Edit parameters of existing device
- Delete device

Edit form looks like the figure below.



The edit form allows to specify following parameters:

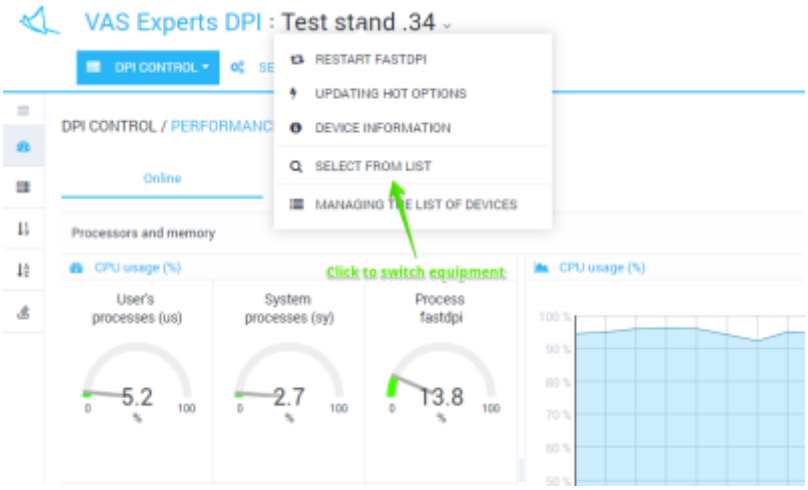
- Device name
- Device IP address used to establish ssh connection
- Port number used to establish ssh connection
- User name
- Password
- Flag, specifying whether the user added to the /etc/sudoers file (allowed to run programs as super user do) or not

Note: For a proper work use a user with sudo rights (defined in the /etc/sudoers file). See section Configuring a connection to DPI

Switching between devices

You can control only one selected device.

To switch to the section "Managing the list of devices" open the "Device_name" menu and press "Select from list".

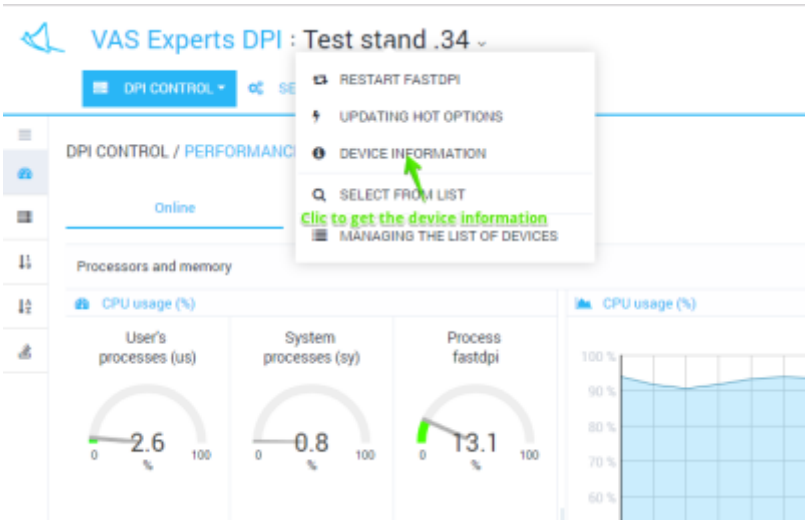


The list (with filtration options) of available devices will be opened.

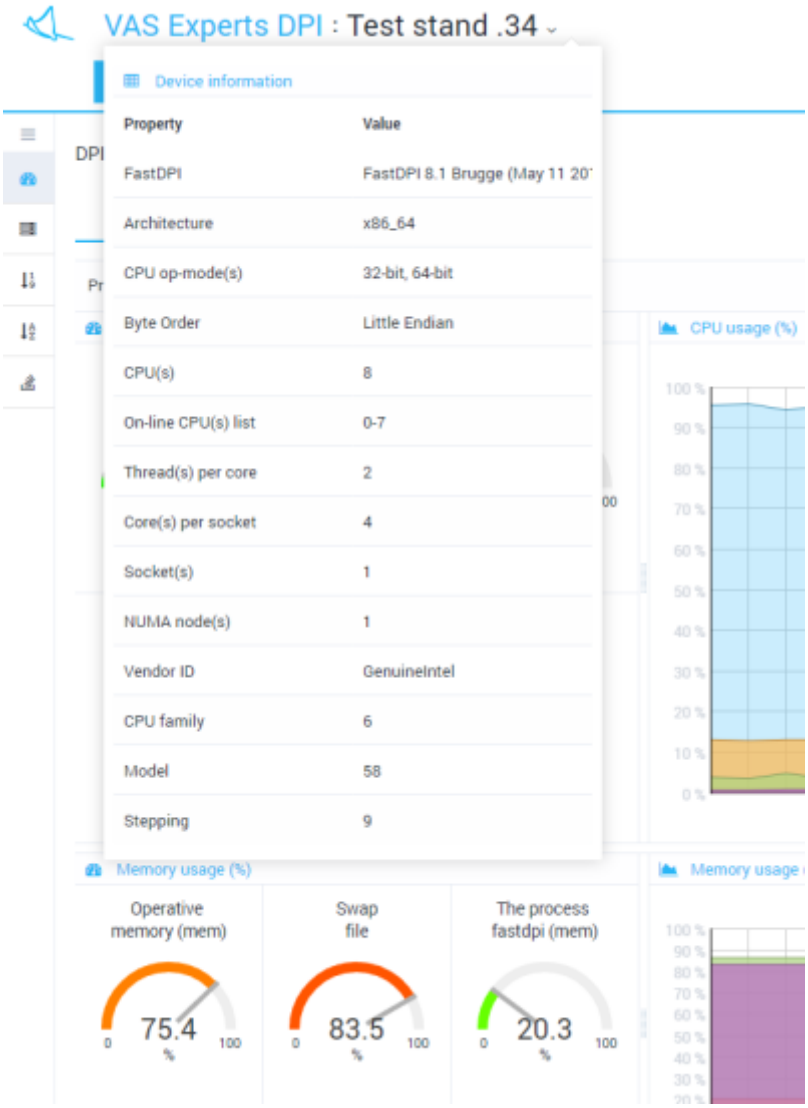
Clicking on a list item causes switching to the device

Device information

For device information open the "Device_name" menu and press the "Device information".



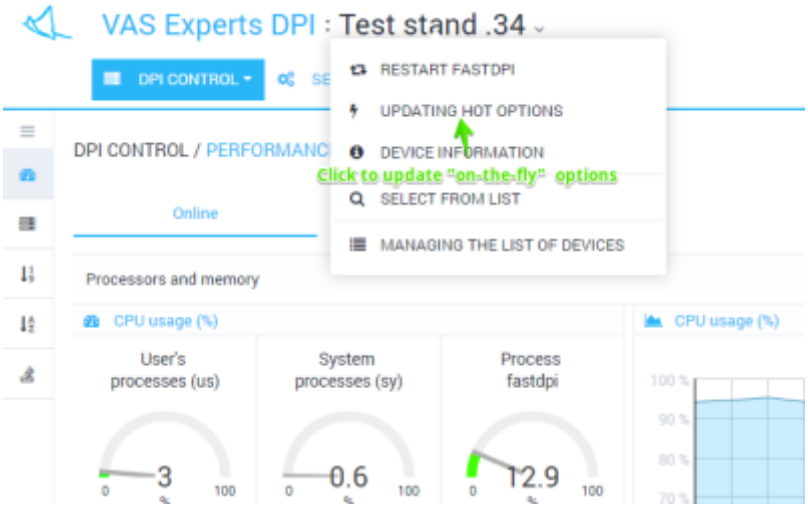
Subwindow "Device information" containing device main characteristics will be opened



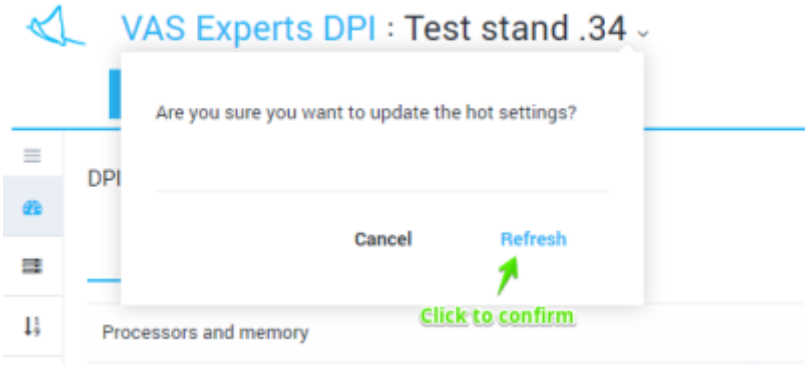
Updating hot options

When you change some options (the so-called "on-the-fly" or "hot" options) in the hardware configuration, it is not necessary to restart the fastdpi service. You can use "Updating hot options".

To update "hot" ("on-the-fly") options open the "Device_name" menu and press "Updating hot options".



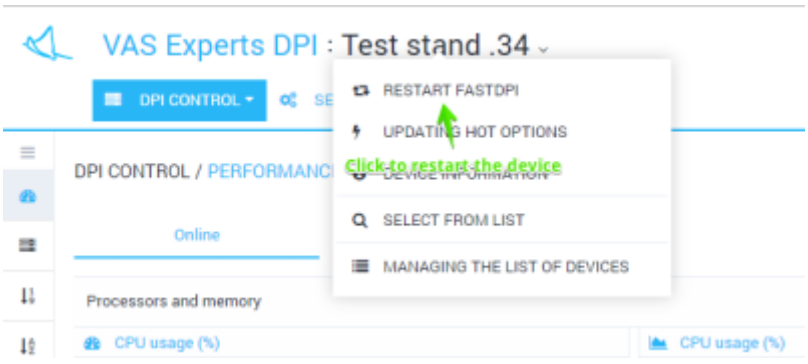
You will be prompted to confirm the operation. Click Refresh.



An operation will be performed. The result will be displayed.

Restart fastdpi

If you change standard (not "on-the-fly") options you need to restart the fastdpi service.



To restart the service, open the "Device_name" menu and click "Restart fastdpi".

A restart will be performed. The result will be displayed.

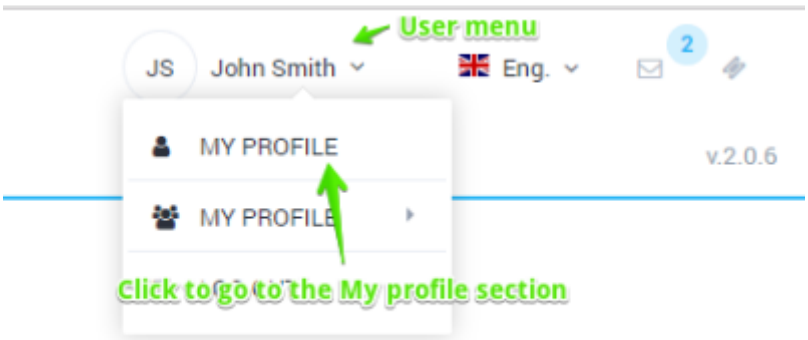
Users and Roles

Section content:

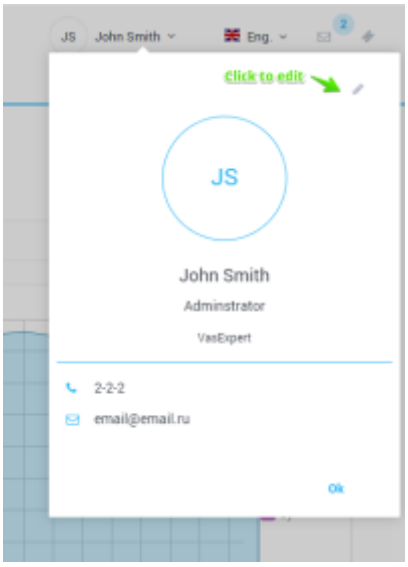
- My profile
- Users list
- Roles

My profile

To open My profile, open the user menu and click on "My profile" user menu.

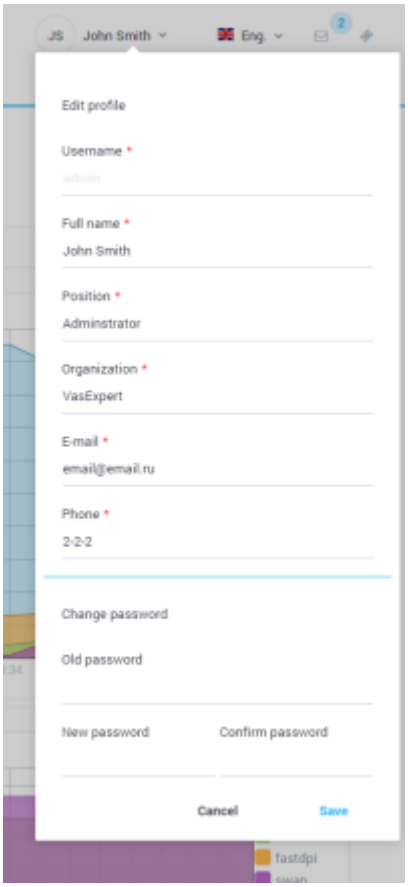


The form looks like the figure below.



Profile editing

To edit a profile on the My profile form, click edit. The editing form will be opened.



The form allows to edit following data:

- Username
- Full name
- Position
- Company
- E-mail

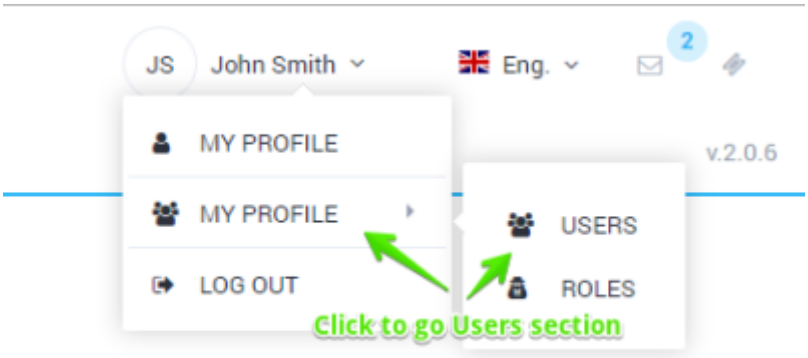
- Phone number

To change the password, enter the old password, new password and confirmation.

To save the changes, click Save.

Users list

To go to the user management section, open the user menu and click Users.

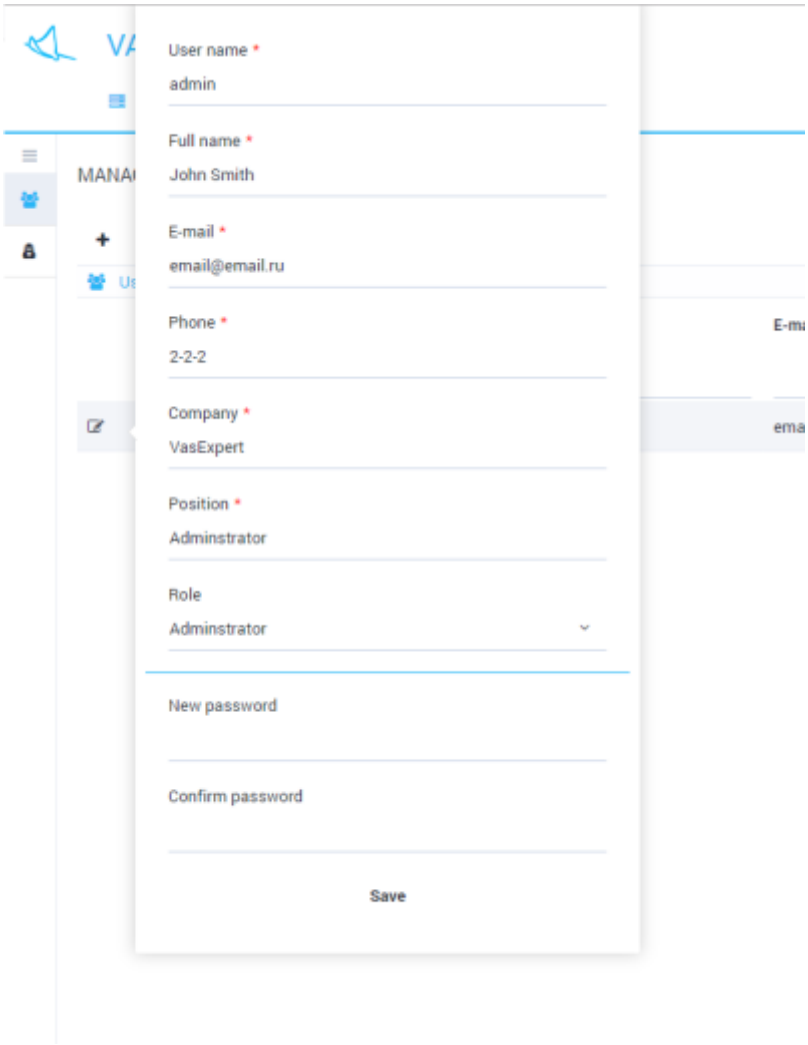


The section looks like the figure below.

The section implements the following features:

- View a list of users with filtering options
- Add a new user
- Edit properties of existing user
- Delete user

The editing form looks like the figure below.



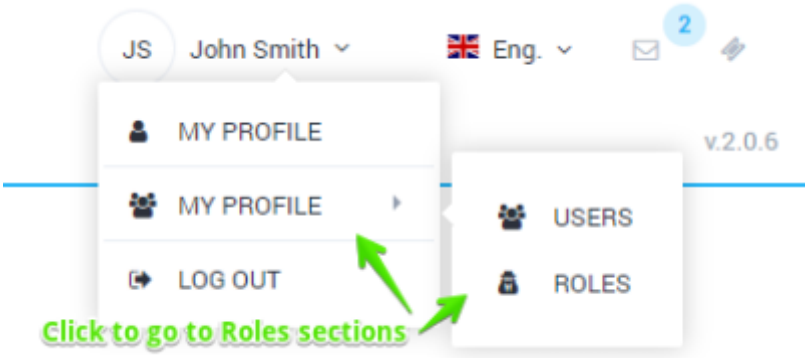
The form allows you to edit the following data:

- User name
- Full name (Last, first and middle names)
- E-mail
- Phone number
- Company
- Position
- Role
- Password

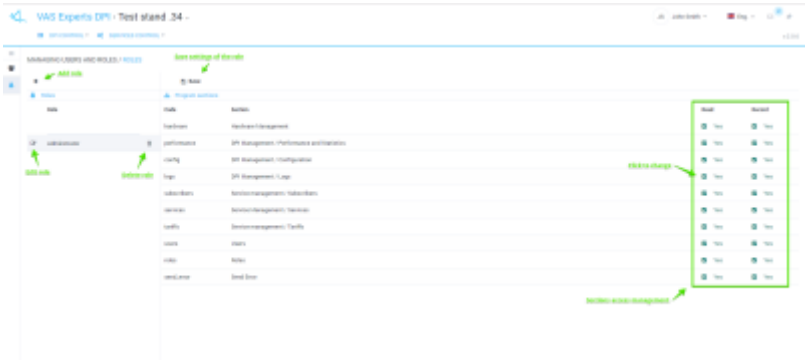
To save the changes, click Save.

Roles

To go to the user management section, open the user menu and in the Users submenu, click Roles.



The section looks like the figure below.



This section implements the following features:

- Add role
- Delete role
- Edit role name
- Manage the access to the sections depending on the role

To save changes, do not forget to click Save.

Performance

To go to the Performance section, open the "DPI CONTROL" menu and click on the "Performance".

The section contains 2 subsections:

- Online - the current state in real time is displayed
- Statistics - the accumulated statistics for the period are displayed

Online

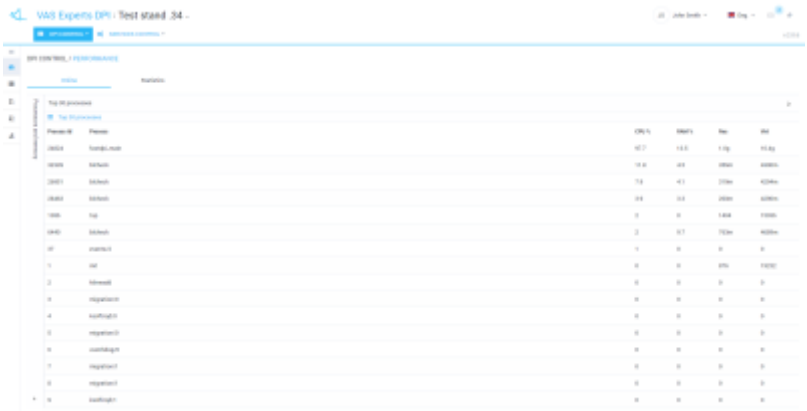
This section shows the current state of DPI performance in real time.

It is possible to switch between 2 tabs:

- Processors and memory - Processors and memory usage are displayed
- Top 30 processes - TOP 30 processes list is displayed

Processors and memory

Top 30 processes



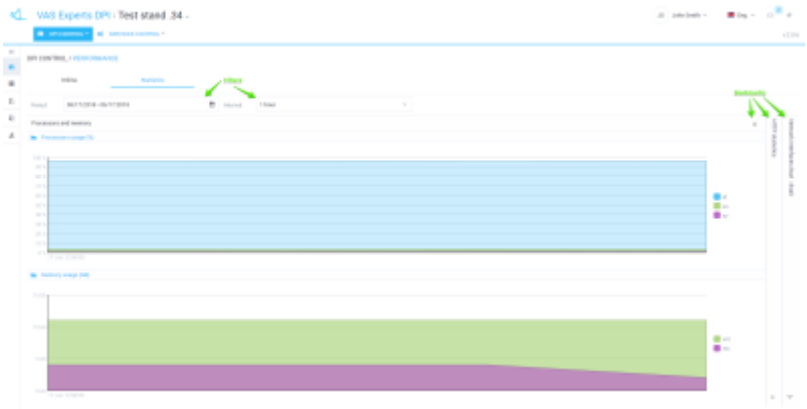
Statistics

This section shows the accumulated statistics for the period. You can change the period and the interval for displaying the data.

It is possible to switch between the following tabs:

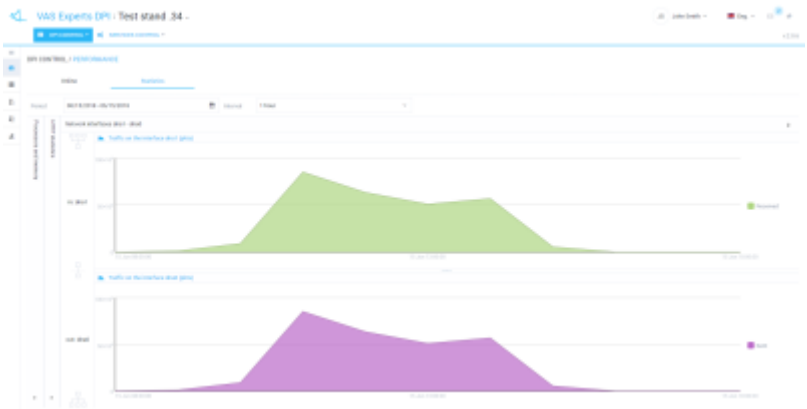
- Processors and memory (statistics) - Processors load and memory usage statistics are displayed
- HTTP statistics - processed and blocked HTTP requests statistics are displayed
- Network interfaces dna0-dnaX - traffic statistics per dna interface

Processors and memory (statistics)



HTTP statistics

Network interfaces dna0-dnaX

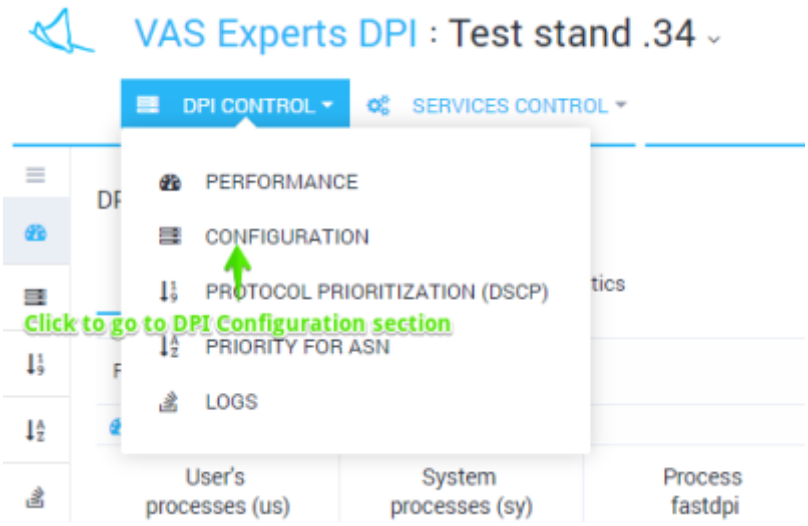


DPI Configuration

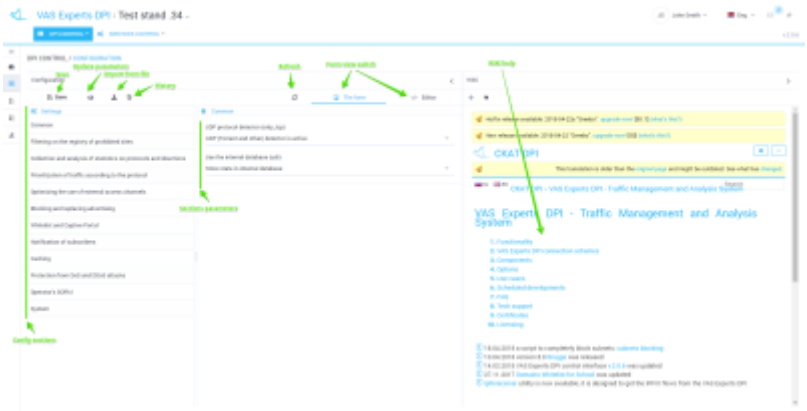
In this section, you can manage the DPI platform settings.

Editing

To go to the Configuration section, open the `DPI CONTROL` menu and click on `CONFIGURATION`.

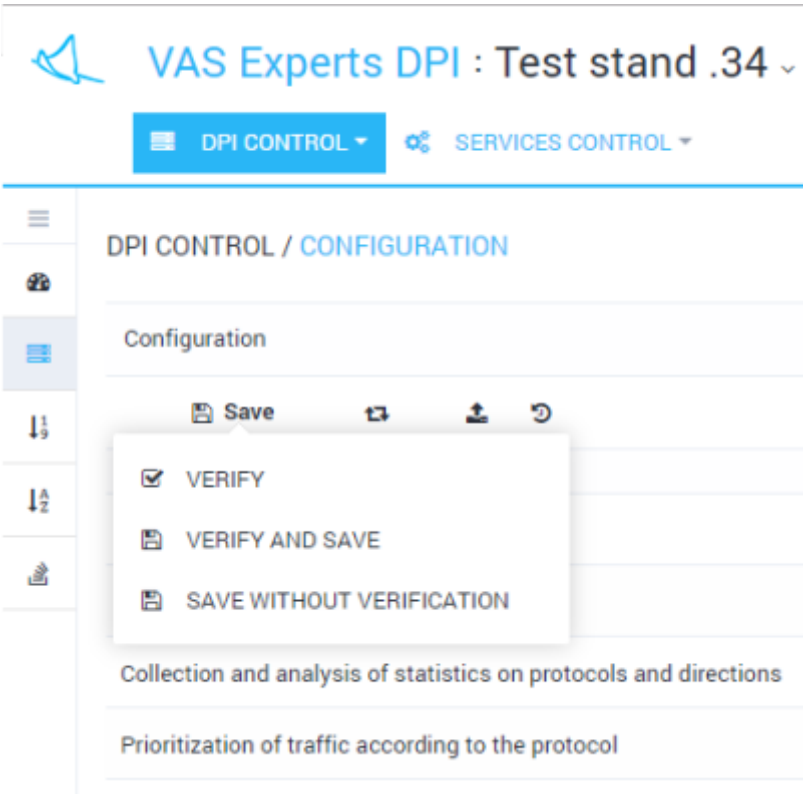


The section looks like the figure below.



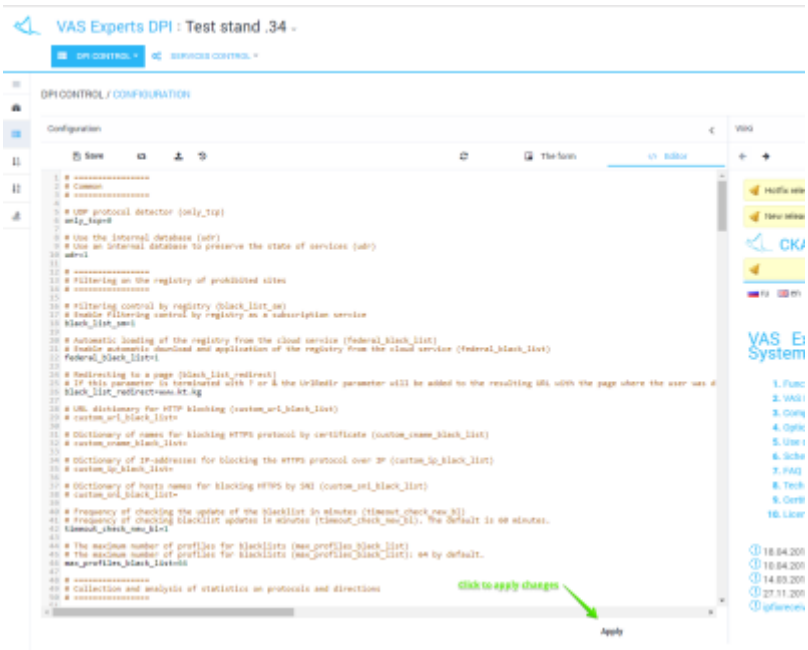
The configuration is divided into sections (the list on the left). For each section corresponding form (centered) and documentation (right) are opened.

Do not forget to save the changes. When you click Save a selection menu appears, there you can check and save the configuration.



Direct editing

It is possible to edit the file directly without using sections and forms. To switch to the direct editing mode, click the `Editor` within the opened `DPI control / Configuration` submenu

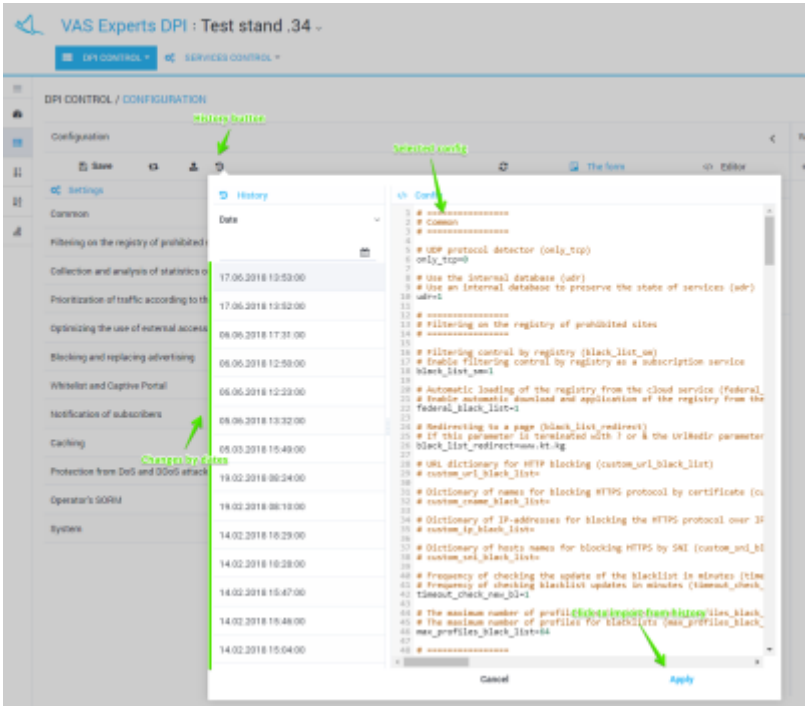


Do not forget to click the `Save` button (after making changes in the editor).

History view and import

To view the history changes in the Configuration section, click the "History" button (Download from

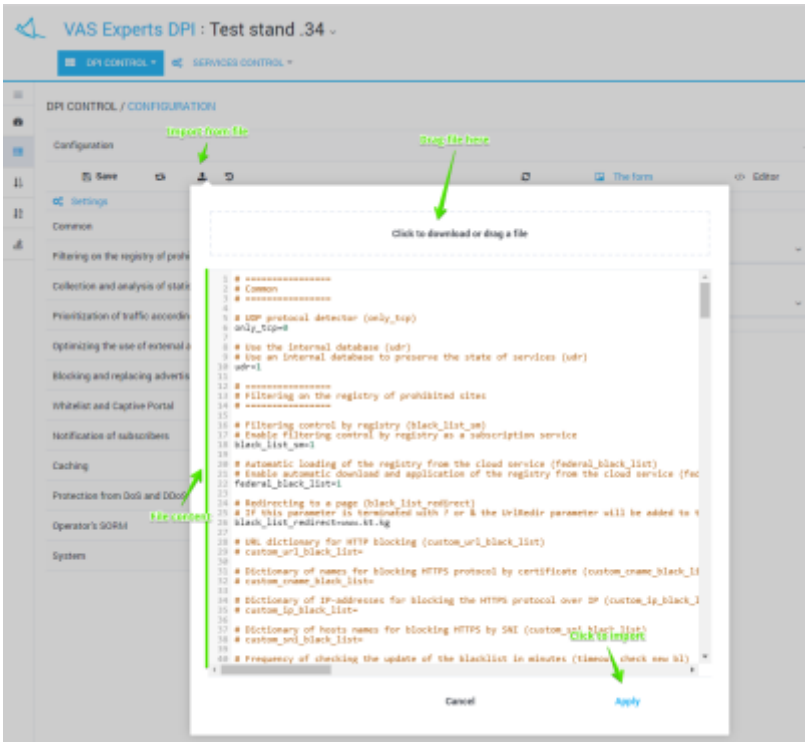
history tooltip). The form will be opened as in the figure below.



It is possible to view and filter a list of changes by date. Select and apply a history change.

Import and view from file

To import from a file click the Import button (Upload from file tooltip) in the Configuration section. The form will opened as in the figure below.



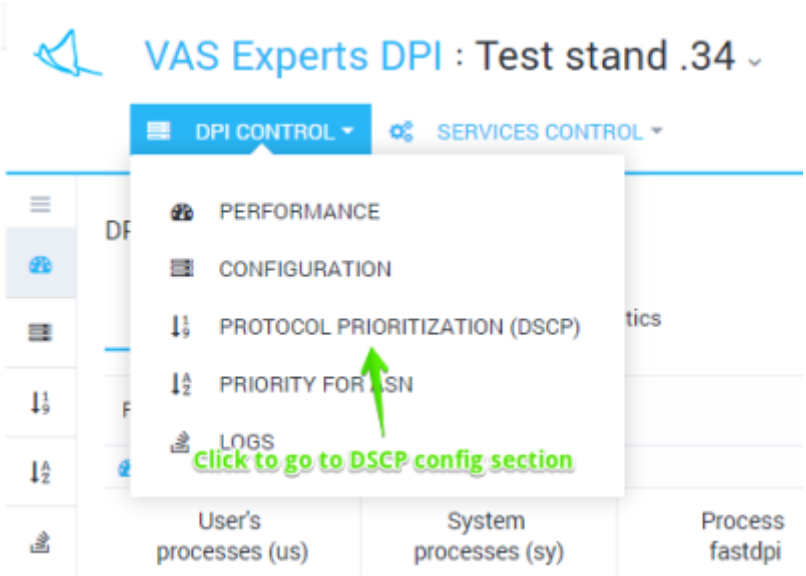
Drag the file to the import area. The contents of the file will be displayed below. Click Apply to import

the contents of the file.

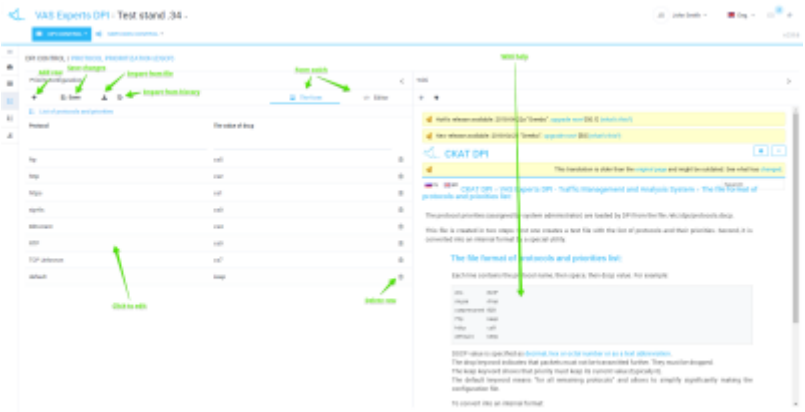
Protocol prioritization (DSCP)

Editing

To go to the corresponding section, open the `DPI CONTROL` menu and click "PROTOCOL PRIORITIZATION (DSCP)".

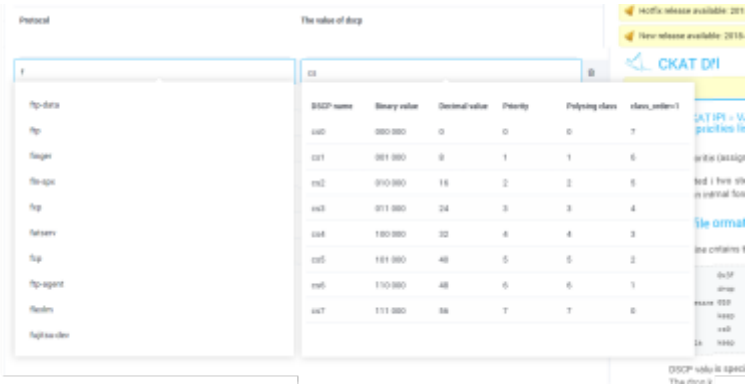


The section looks like the figure below.



You can view and filter the list of protocols along with its corresponding priorities and add, delete or edit in this section.

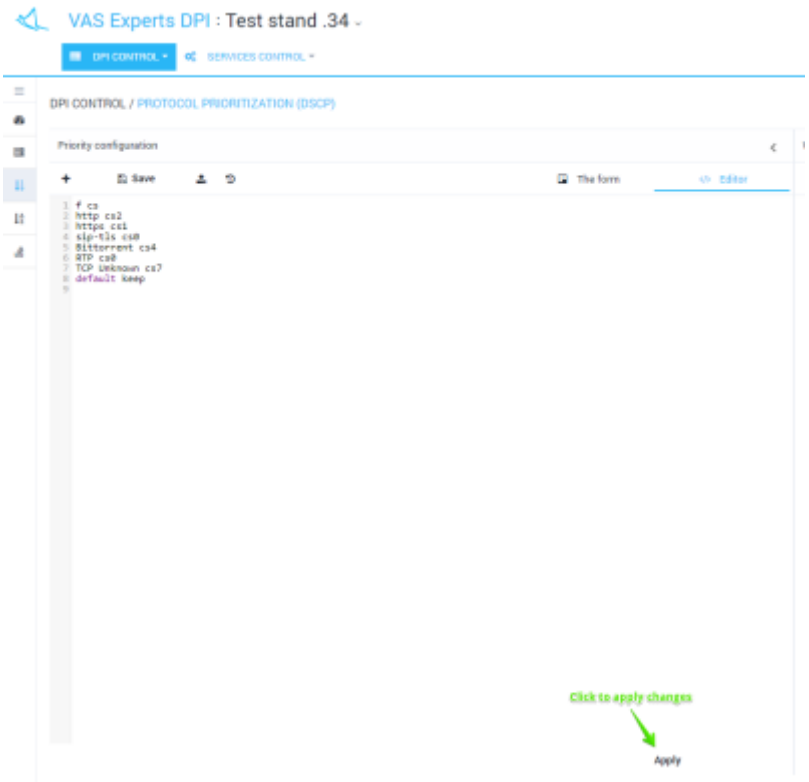
Editing is done in tabular form. To edit, just click on the line. When filling out the protocol values and dscp, drop-down lists of hints appear.



Do not forget to click Save after making changes.

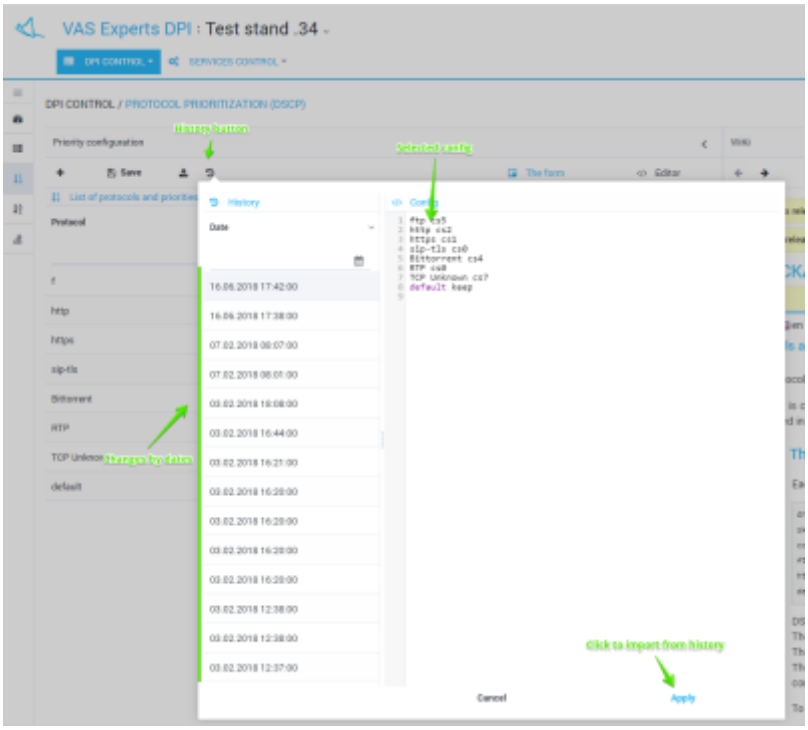
Direct editing

It is possible to edit the file directly without using a table form. To switch to the direct editing mode, click the "Editor".



History view and import

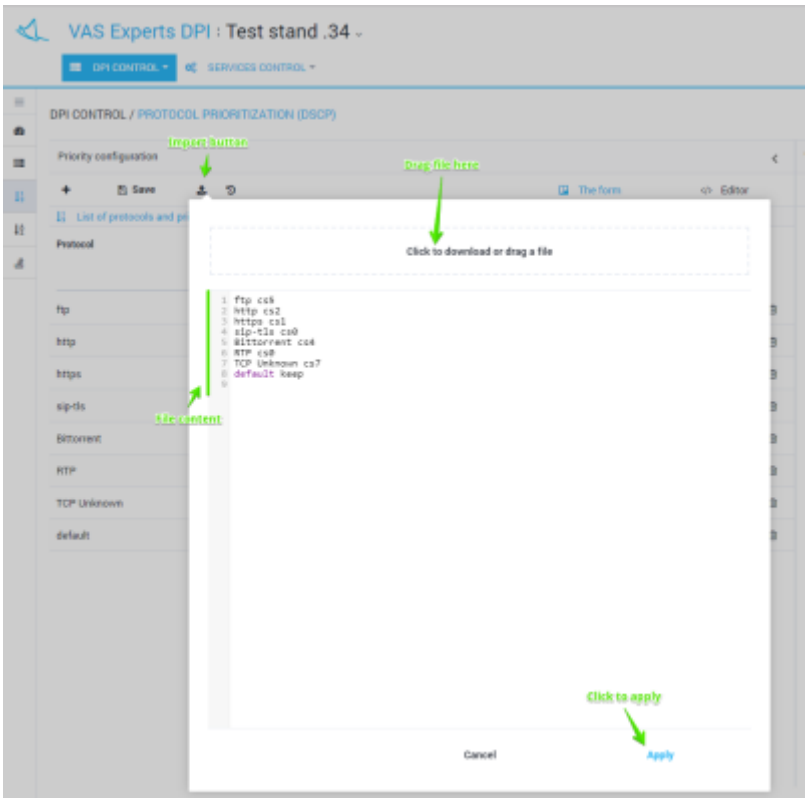
To view the history changes, click the History button. The form will be opened as in the figure below.



It is possible to view and filter a list of changes by date. Select and apply a a history change.

Import and view from a file

To import from a file, click the "Import" button. The form will be opened as in the figure below.

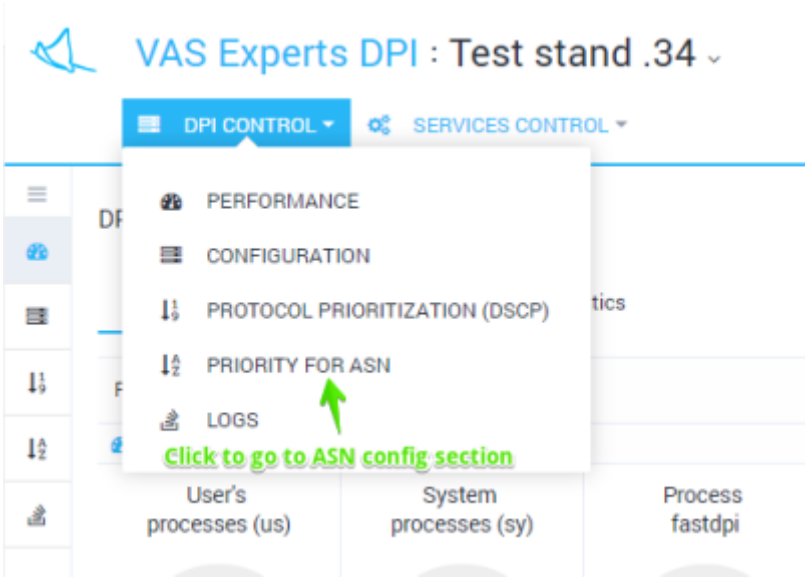


Drag the file to the import zone. The contents of the file will be displayed below. Click Apply to import the contents of the file.

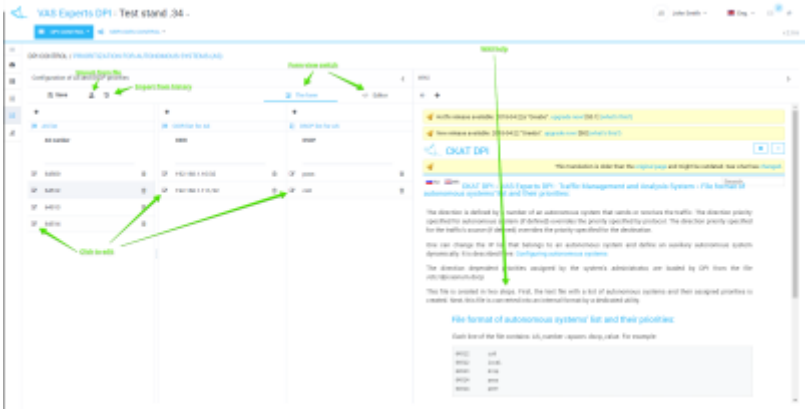
PRIORITY FOR ASN

Editing

To go to the section, open the "DPI CONTROL" menu and click "PRIORITY FOR ASN".



The section looks like the figure below.



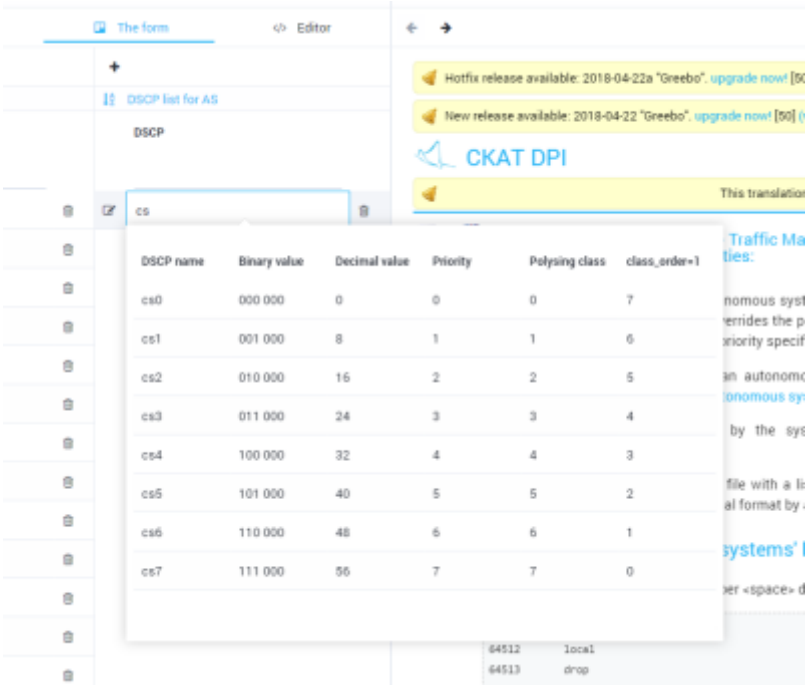
In this section, you can view and filter the list of AS (autonomous systems) and its corresponding priorities and CIDR, add, delete or edit them.

Editing is done in tabular form. To edit, double-click on the line.

Editing is conditionally divided into 3 stages:

- The AS list is filled in/edited (left column)
- The CIDR list is filled (central column) for each AS
- The DSCP list for AS is filled (right column) for each AS

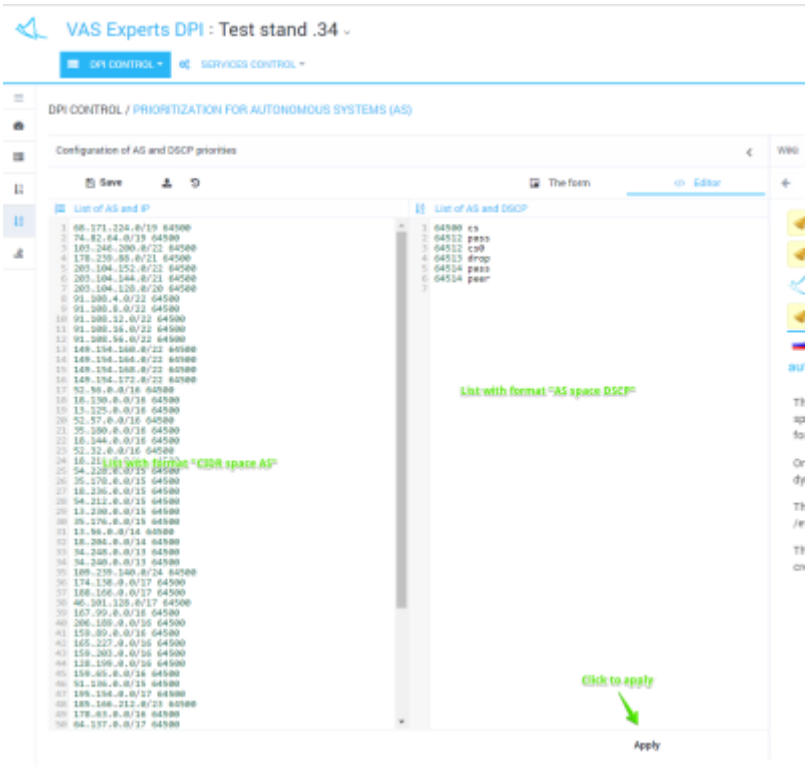
When populating the dscp values, a drop-down prompt appears.



Do not forget to click Save after making changes.

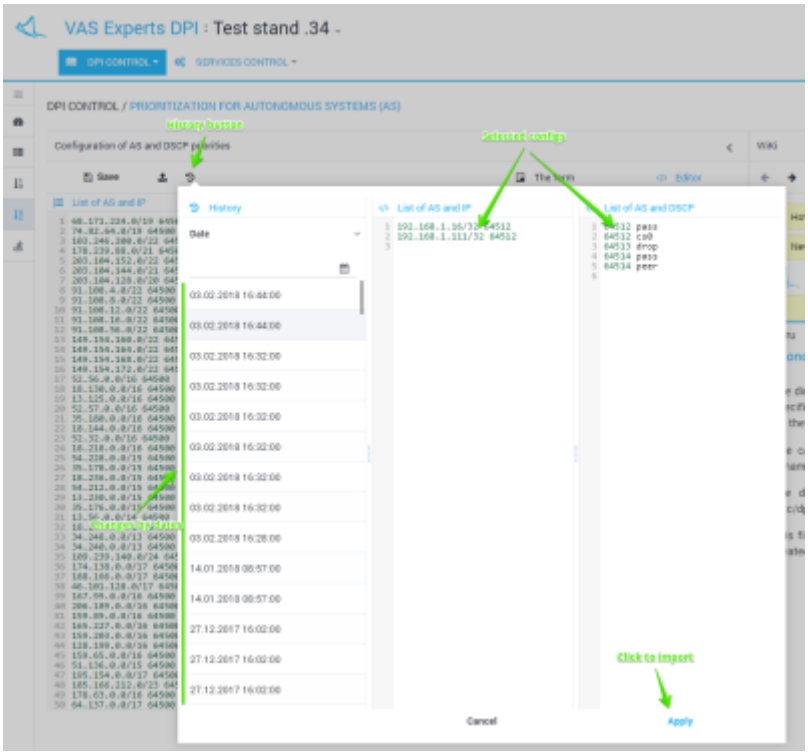
Direct editing

It is possible to edit the file directly without using tabular forms. To switch to the direct editing mode, click "Editor".



View and import from history

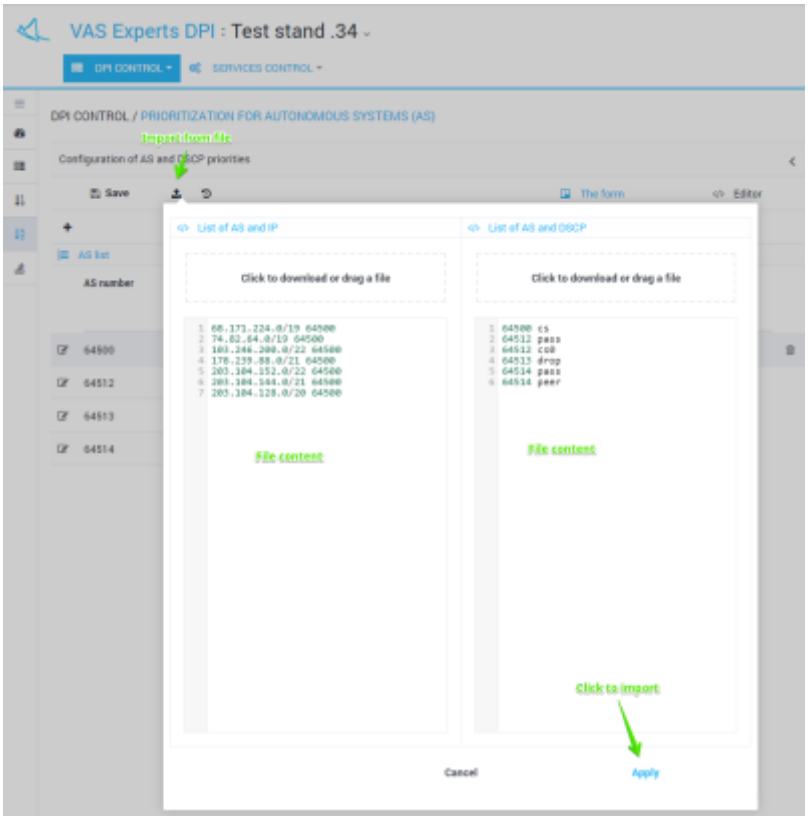
To view the history changes, click the "History" button. The form will be opened as in the figure below.



It is possible to view and filter by date a list of changes. Select and apply a change from the history.

Import and view from a file

To import from a file, click the "Import" button. The form will be opened as in the figure below.



Drag the file to the import zone. The contents of the file will be displayed below. Click Apply to import the contents of the file.

DPI Logs

To go to the section, open the `DPI CONTROL` menu and click `LOGS`

The section looks like the figure below.

The last 1000 log lines are displayed:

- Information messages and errors log
- Statistics log

It is possible to export log files entirely.

Subscribers and services

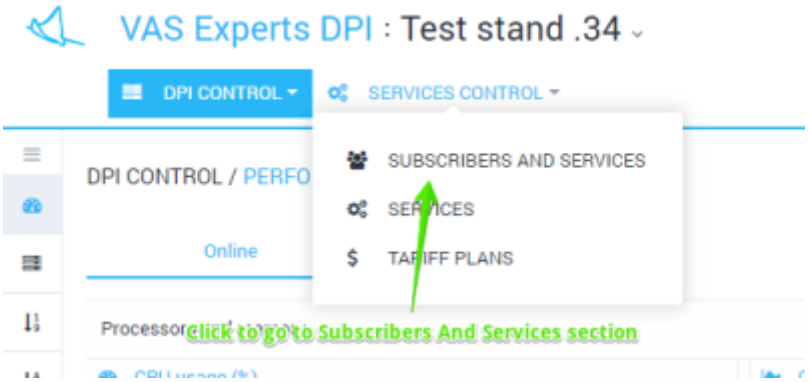
This section `SERVICES CONTROL / SUBSCRIBERS AND SERVICES` allows you to manage the list of all subscribers processed by DPI device and bind services to them (so far without profiles - this feature is currently under development).

Lists synchronization is performed every 30 minutes on a schedule.

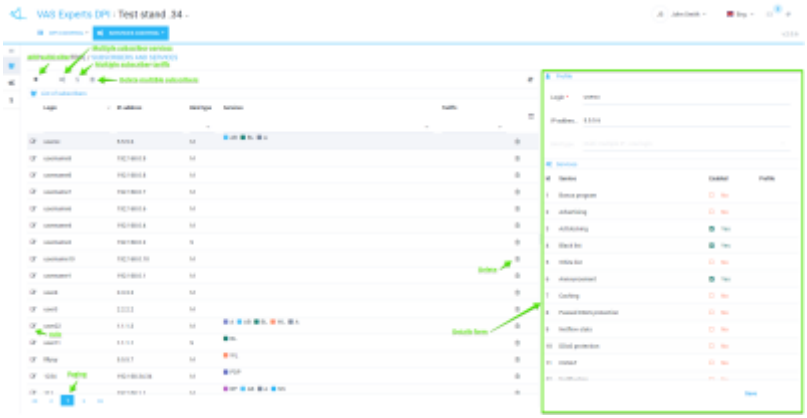
It is possible to add subscribers with the following connection types:

- `Single`: one login (subscriber) is assigned only one ip address
- `Multi`: one login (subscriber) can be assigned several ip addresses, also it can be assigned a range of ip addresses and CIDR
- `Without bind`: the subscriber does not have a login, and the services are assigned using the ip address

To go to the section, open the `SERVICES CONTROL` menu and click on "SUBSCRIBERS AND SERVICES".



The section will be opened as on the figure below.

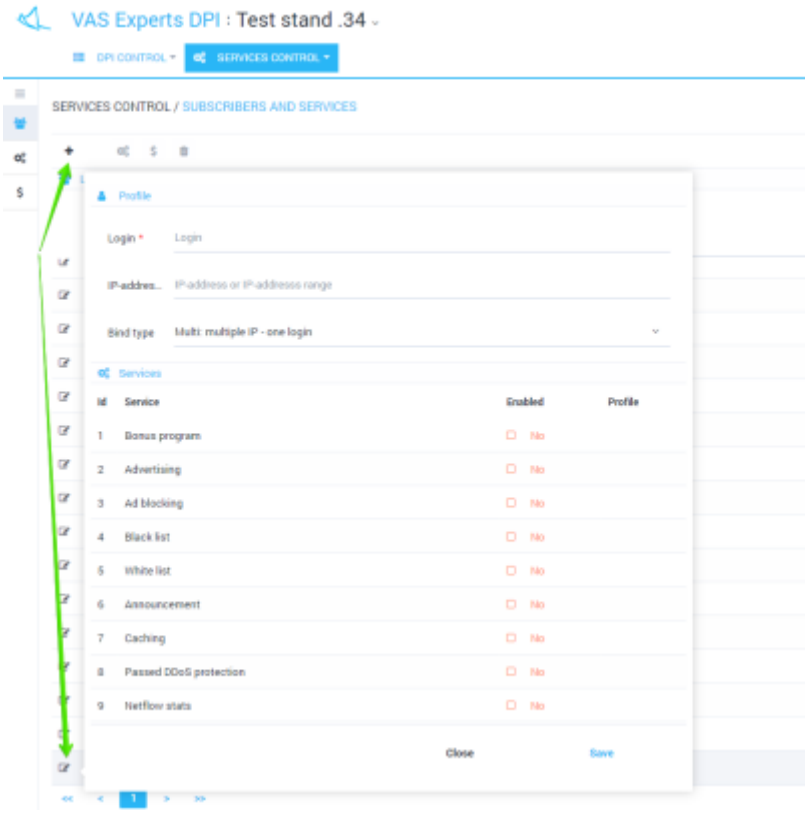


The section has the following features:

- View the list of subscribers and activated services with filtering
- Details by subscriber
- Add / edit subscriber properties
- Enable / disable services
- Perform group operations

Adding a new subscriber

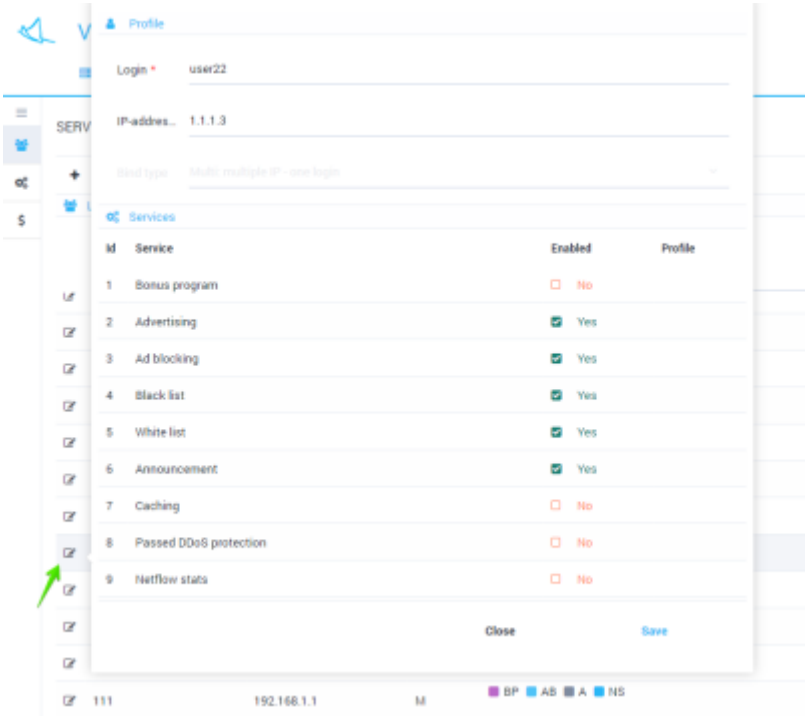
Click the Add button. At the bottom of the table, a new line will be added and the form will be opened as in the figure below.



Select the connection type, fill in the Login and IP address fields. Activate the necessary services. Click Save.

Editing a subscriber

You can edit the subscriber either by using the detailization form (on the right) or by using another separate form, if you click on the Edit icon opposite to the subscriber.



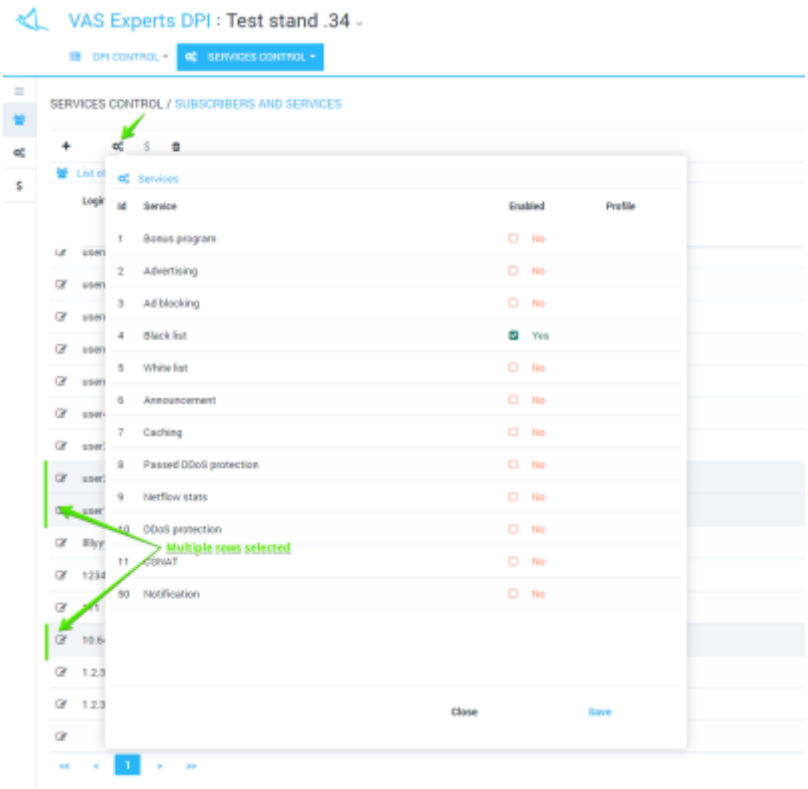
You can change the subscriber login and IP address. For the subscriber using the "Without bind" connection type you can change only the IP address. Services can be enabled or disabled.

Note: if the subscriber has a "Multi" connection type, then enabling and disabling of services will be performed for all of the subscriber's IP addresses.

Group-wide services enabling/disabling

It is possible to enable and disable services to a group of subscribers at once. To do this, select several lines (with ctrl or shift) and click the Services button.

A form will be opened, where you can activate and deactivate services for the selected subscribers.



Services

Is under development

Tariff plans

Is under development

QoE Analytics

This section appeared in version 2.1.0.

Do not forget to configure connection to QoE Stor.

The main section is divided into several subsections for ease of use.

QoE DASHBOARD

QoE NETFLOW

QoE CLICKSTREAM

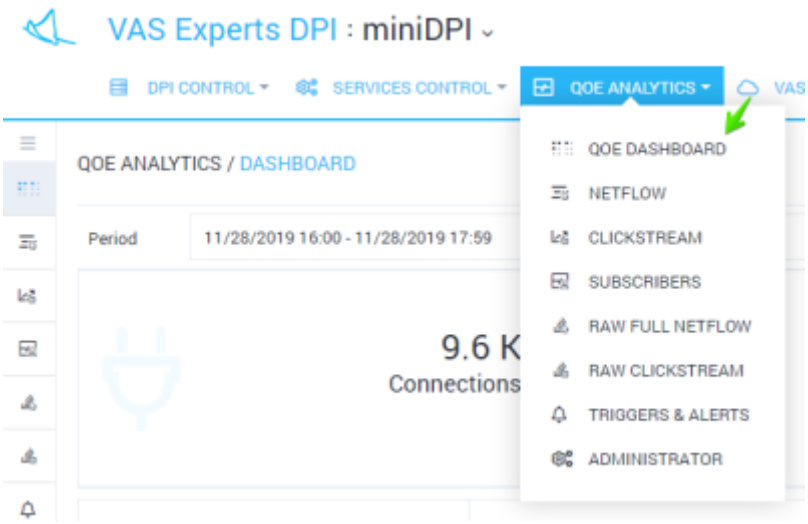
QoE SUBSCRIBERS

Sections above contain lists of reports (tabular and graphical) and filters. Each report allows data analysis in various forms.

QoE DASHBOARD

The section contains all of the report widgets available within the system. Widgets can be moved, added (by moving it from the right side to the left), and deleted.

To switch to the section, open the QoE ANALYTICS menu and click QoE DASHBOARD.

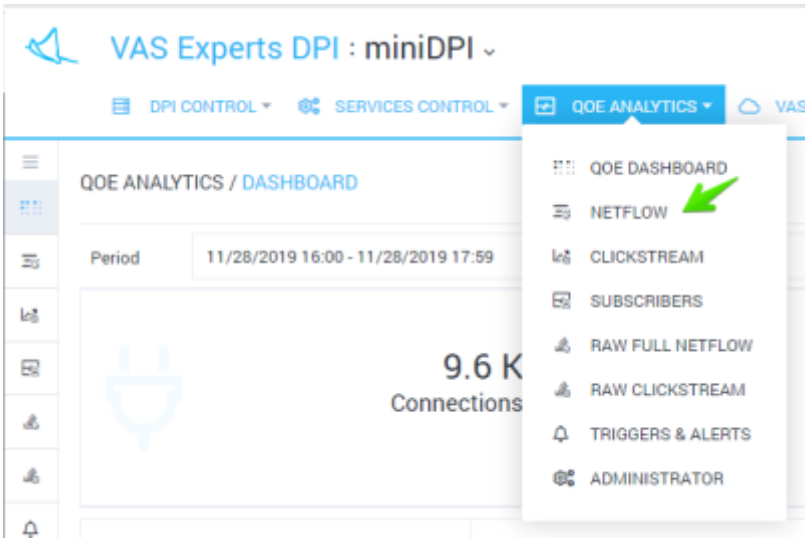


The section will be opened like the figure below.

QoE NETFLOW

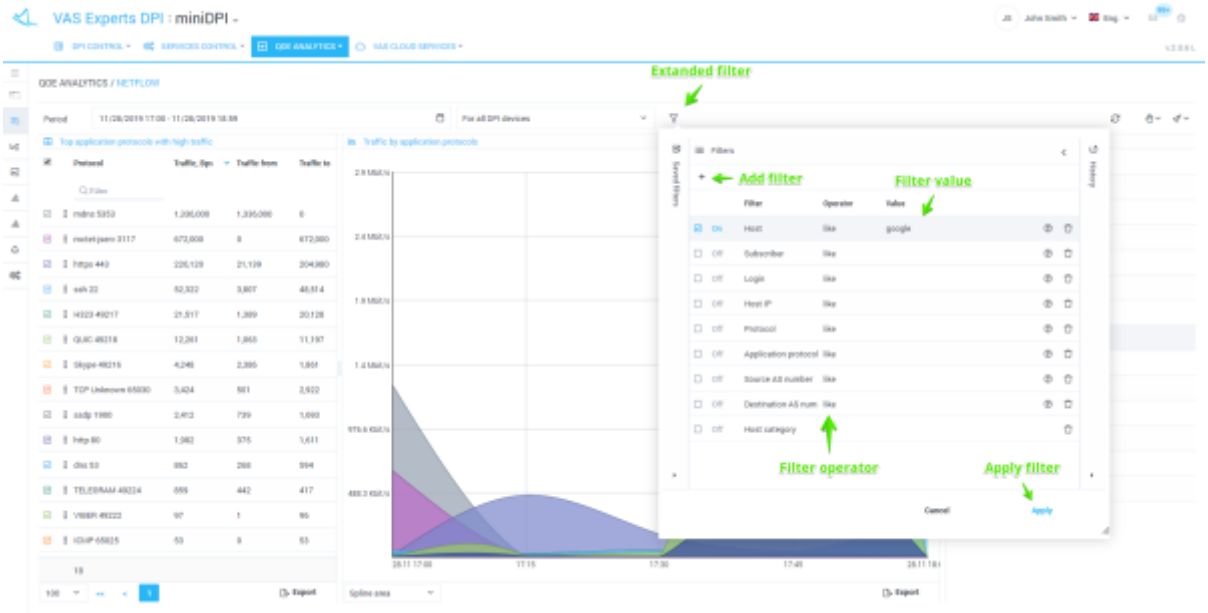
This section contains reports for netflow analysis.

To switch to the section, open the QoE ANALYTICS menu and click NETFLOW



The section will be opened like the figure below.

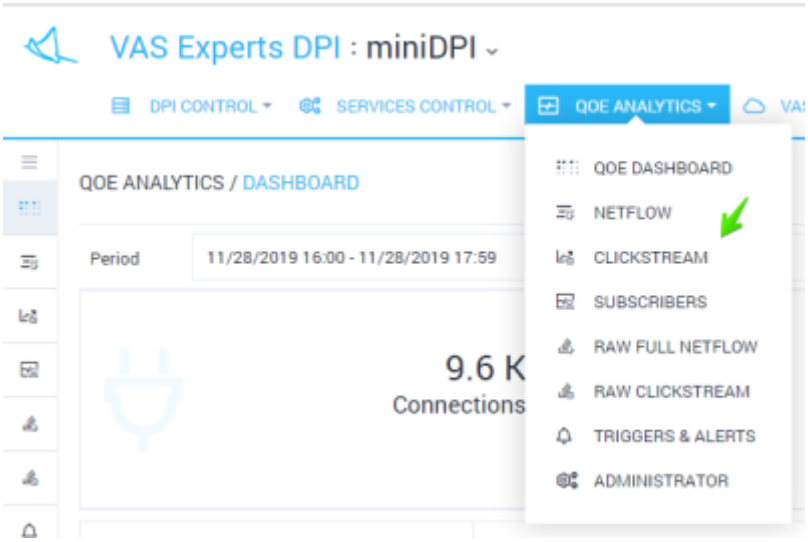
You can set filters for advanced analysis.



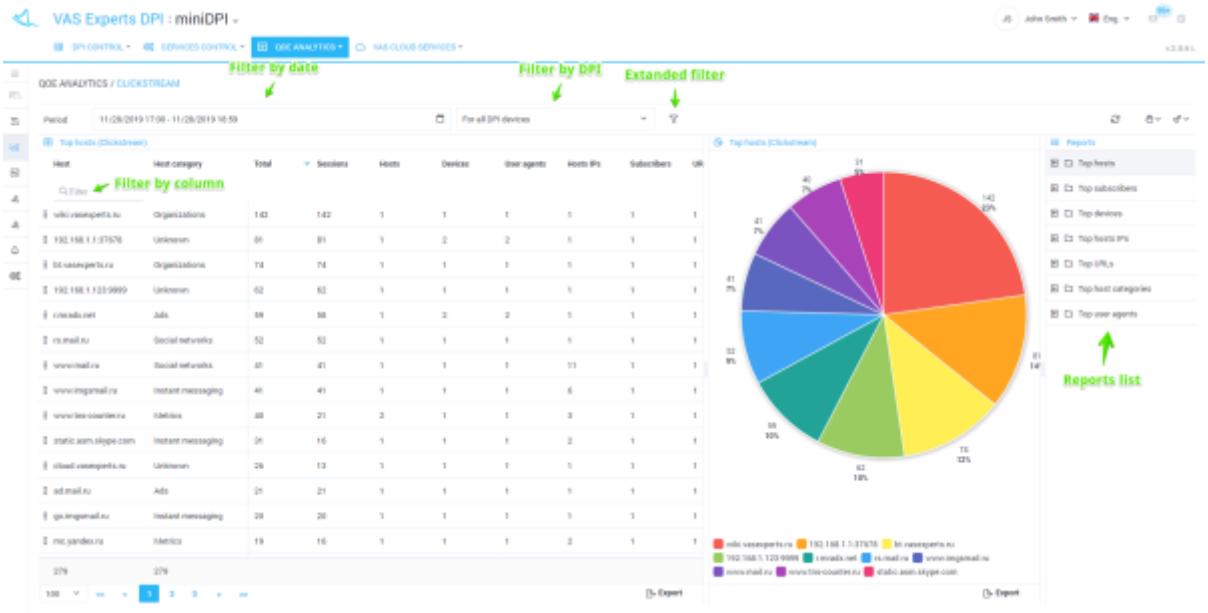
QoE CLICKSTREAM

This section contains reports for clickstream analysis.

To switch to the section, open the `QoE ANALYTICS` menu and click `CLICKSTREAM`



The section will be opened like the figure below.

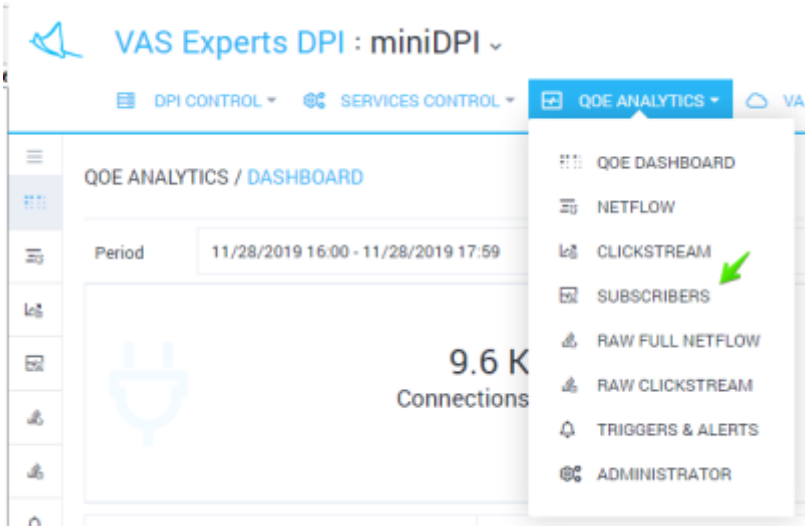


QoE SUBSCRIBERS

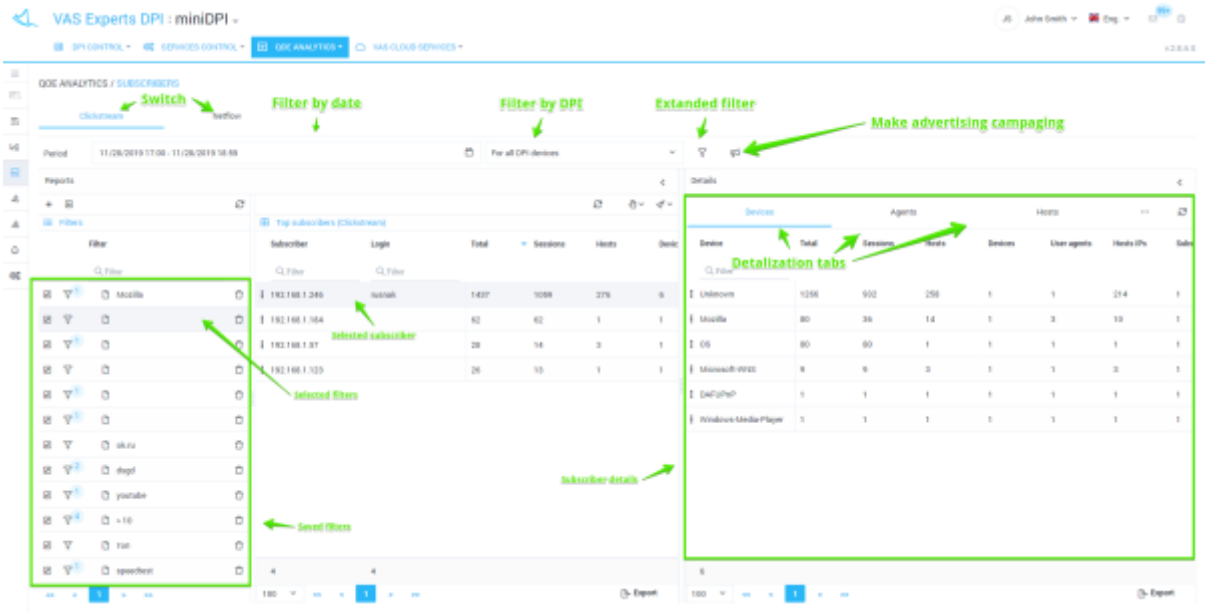
This section contains reports for netflow and clickstream analysis in terms of subscriber properties.

The feature of the section is that you can create a custom list of named filters (search patterns): for example, search for subscribers visiting competitors' web-sites; search for subscribers using IPTV or VoIP; search for Internet speed measurements performed by subscribers and so on.

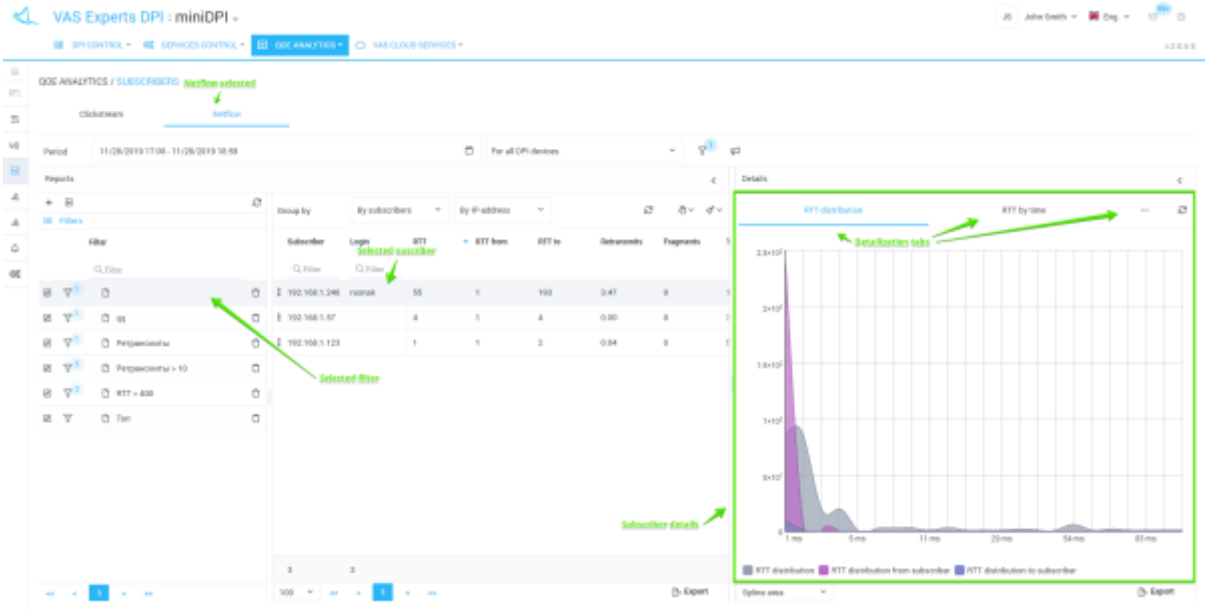
To switch to the section, open the `QoE ANALYTICS` menu and click `SUBSCRIBERS`



The section will be opened like the figure below.



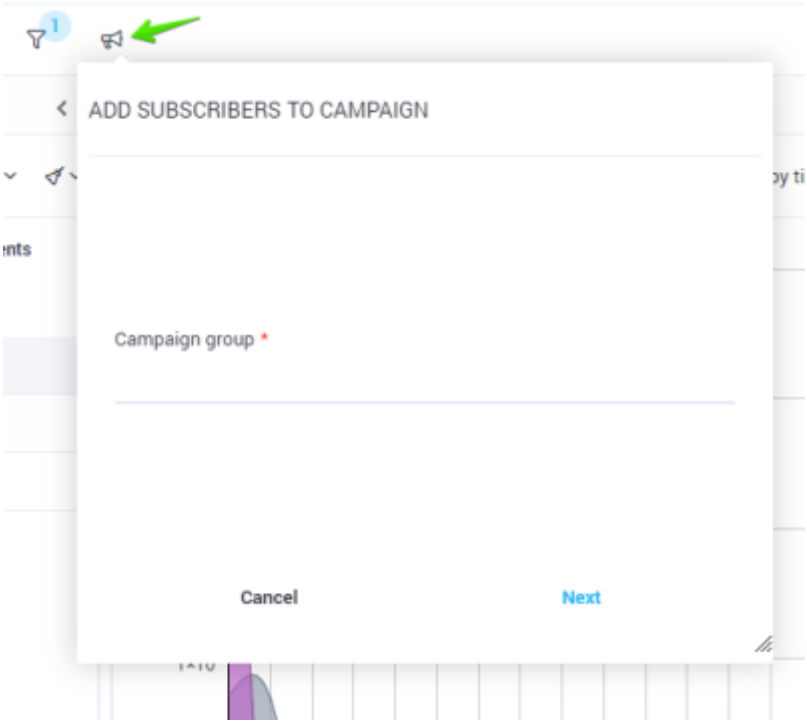
To analyze subscribers in terms of netflow, please open `Netflow` tab.



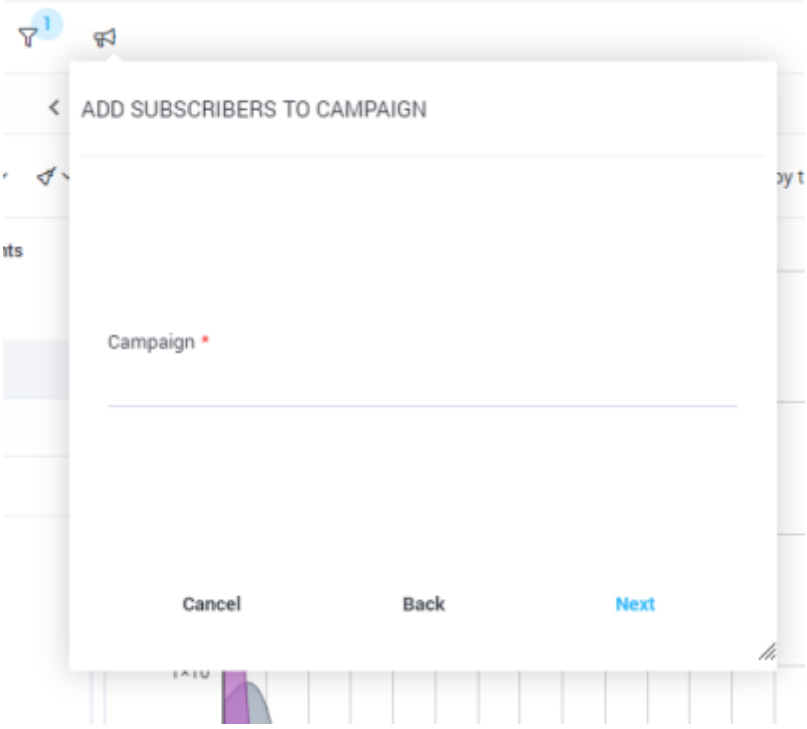
QoE subscribers exporting to the advertising campaigns

In the “QoE Subscribers” section there is a feature allowing to create an advertising campaign and export a list of filtered subscribers.

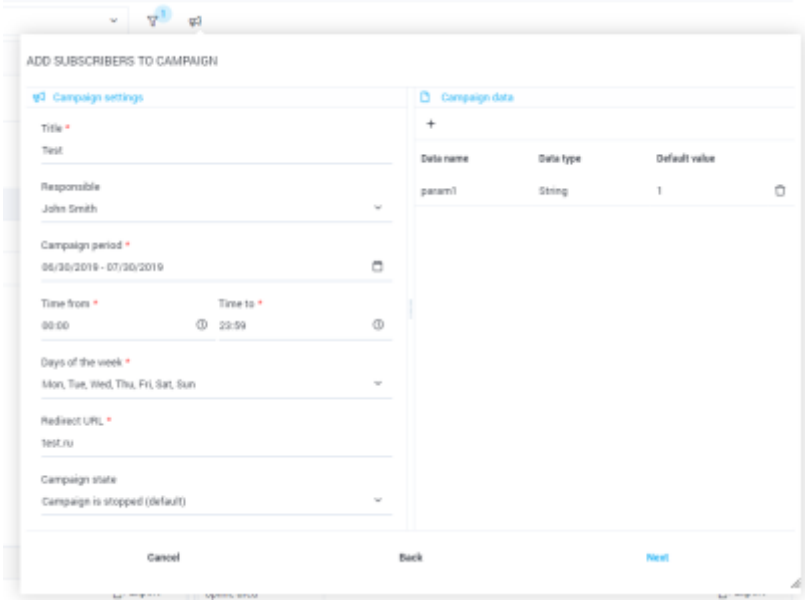
Go to the QoE `ANALYTICS/SUBSCRIBERS` section. Select a filter. Click the “Add Subscribers to Camping” button.



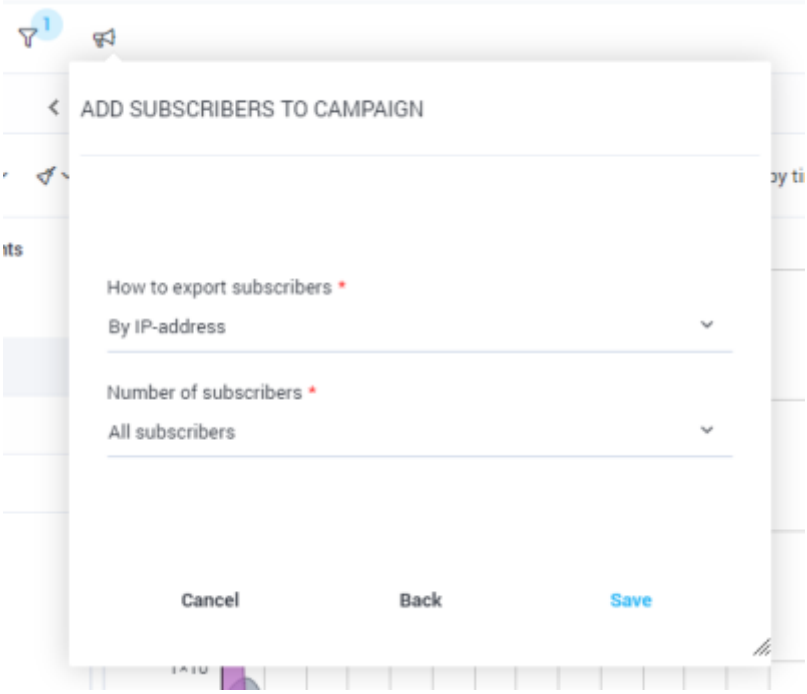
Enter the campaign profile (a new one will be created if there is no profile yet). At the next step, enter the campaign name (you will create a new one if there is no campaign yet).



At the next step, the campaign edit form will be opened. You can fill in or edit data.



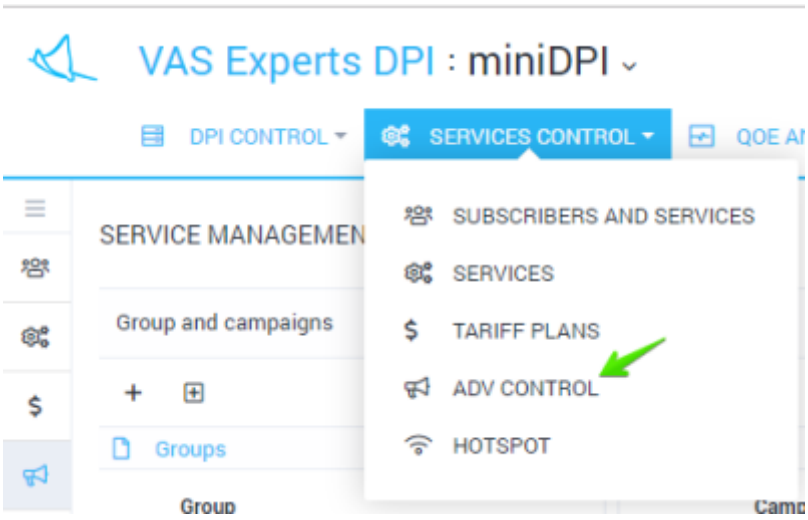
Finally, you need to choose a way to export subscribers: either by the IP address or login. If necessary, limit the number of subscribers.



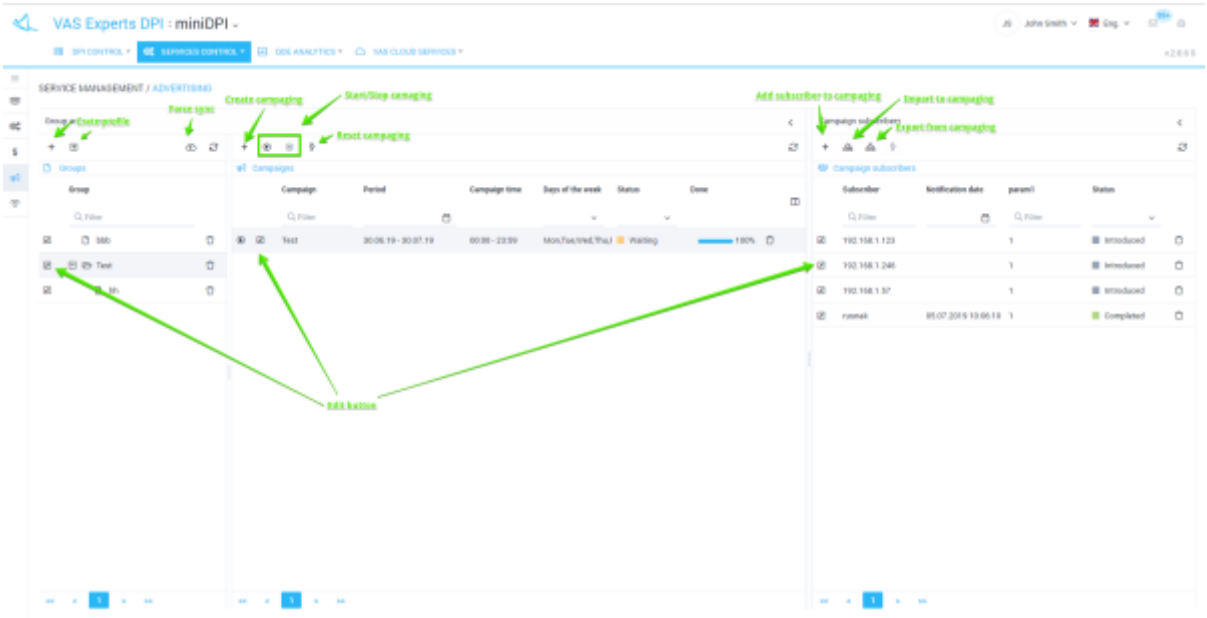
Advertising control

This section appeared in version 2.1.0.

To switch to the section, open the menu `SERVICES CONTROL` and click `ADV CONTROL`.

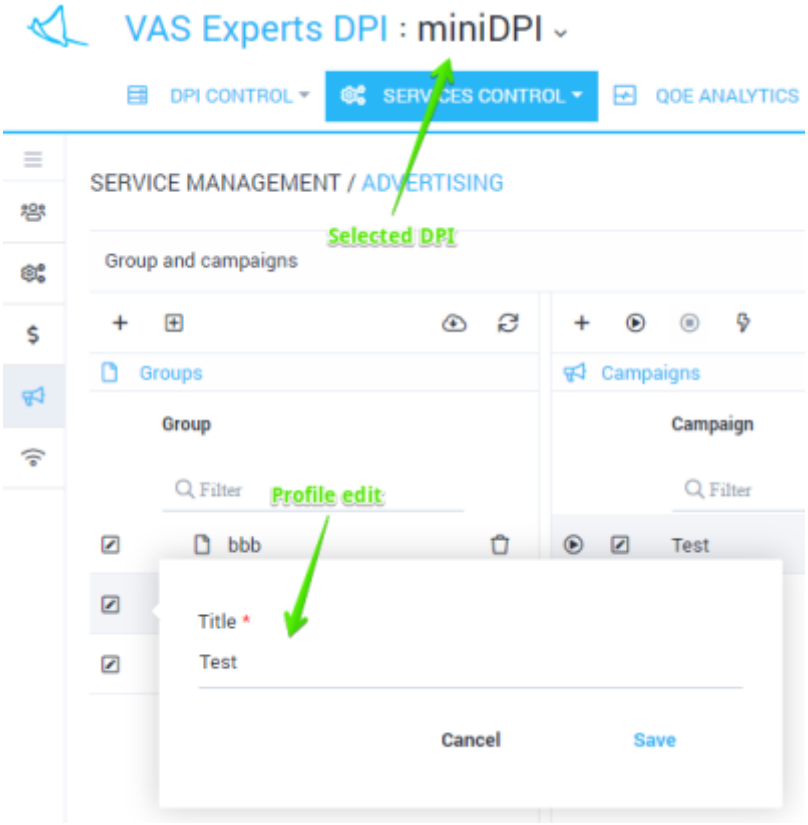


The section looks like the figure below.



Advertisement campaign profiles

Here you can create profiles for combining several adv campaigns, as well as edit and delete them. The form for creating/editing advertising campaigns profiles is shown in the figure below.

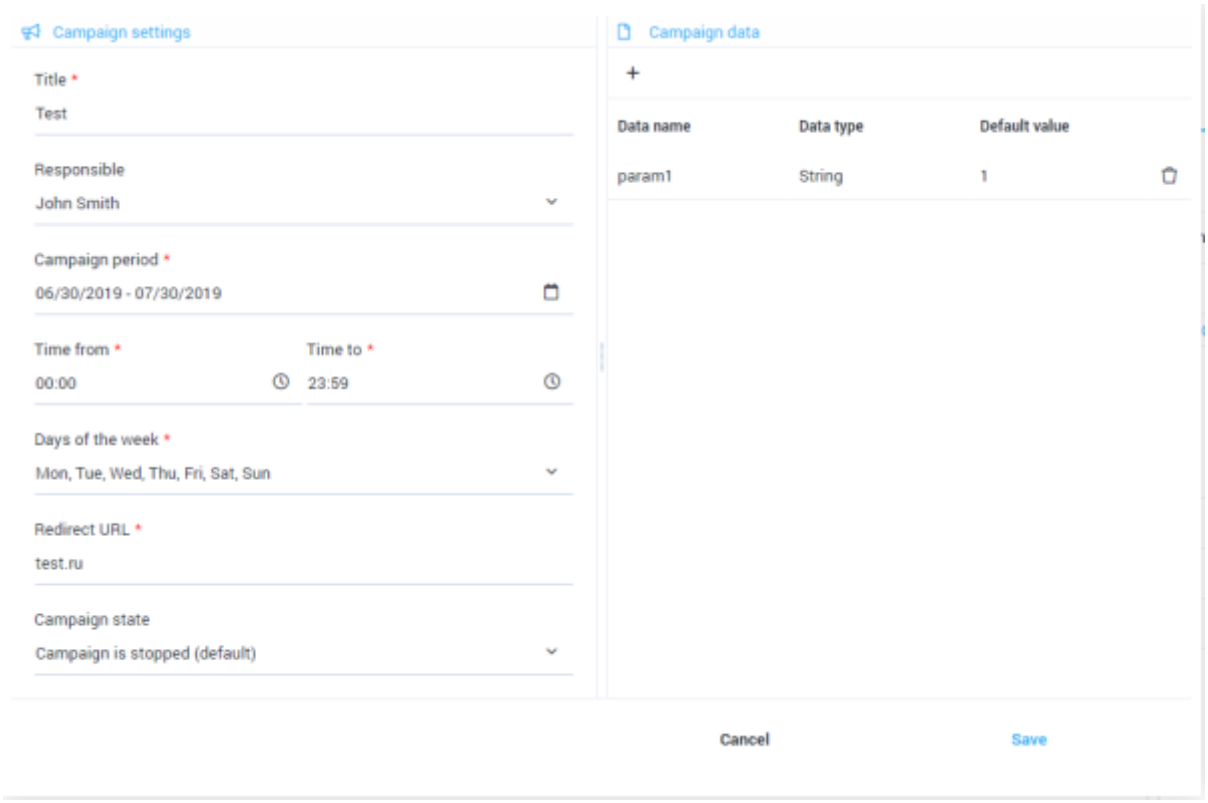


You can control advertising campaigns for only one selected device.

Advertisement campaigns

Here you can create advertising campaigns for the selected advertising campaign profile. Along with deleting, starting and stoping the previously created advertising campaigns.

The form allowing to create/edit an advertising campaign is shown in the figure below.



The form allows you to fill in the following parameters:

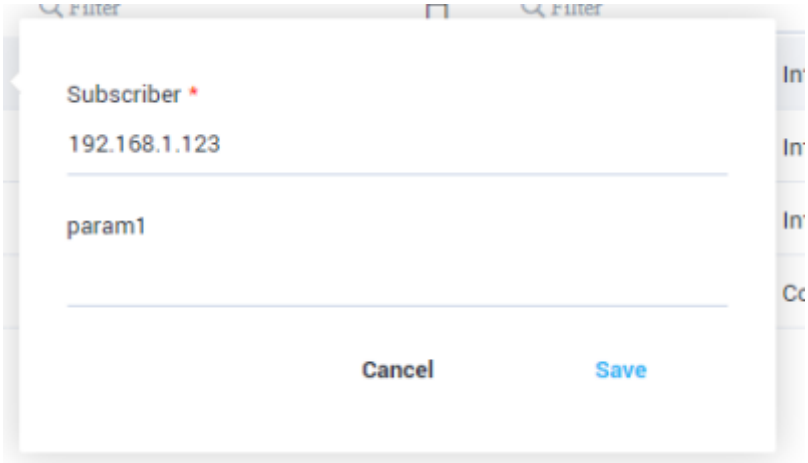
- Campaign Name
- User responsible for the campaign
- Campaign Period
- Campaign time
- Days of the week
- Redirect URL
- Campaign status (It is stopped by default. In order to start a campaign, you should select the "Started" option in the form or use the start/stop buttons shown in the "Section" figure)
- Campaign data (campaign parameters) used to generate a redirection URL for subscribers added to the campaign

Subscribers of advertisement campaigns

Here you can manage the list of subscribers and the campaign parameters set for them (if they were added during the creation/editing of an advertising campaign) for the selected advertising campaign.

Adding/editing subscriber in advertising campaign

The form for editing the subscriber in advertising campaign is shown in the figure below.

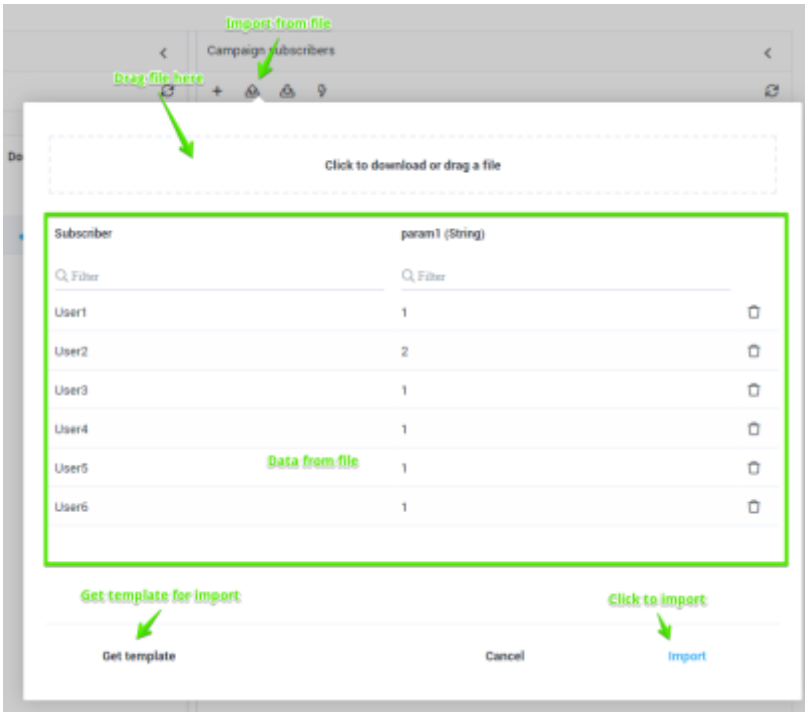


The form allows you to fill in the following parameters:

- Subscriber - login or subscriber IP address
- The values of the advertising campaign parameters for this subscriber (if they were added when creating/editing an advertising campaign). In case you leave these fields blank, the default values specified while creating/editing an advertising campaign will be set.

Importing the subscribers to the adv campaign from file

The form of importing subscribers to an advertising campaign from a file is shown in the figure below.



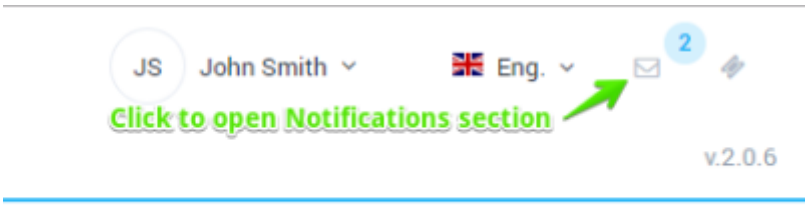
Before importing subscribers and their data into a campaign, it is recommended to download a template Excel file for this campaign (the download button is shown in the figure) in order to make sure that:

- The first column of the table is the name of the account (login) or the subscriber IP address
- The names of the parameters of the advertising campaign in the file matches to specified ones

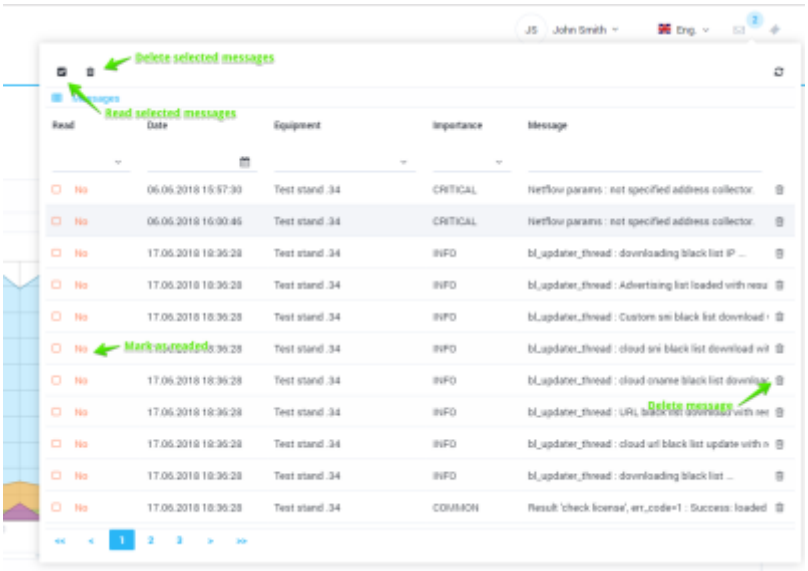
in the campaign

Notifications

In the upper right corner of the program window there is a Message button. Above the button the indicator of CRITICAL messages number is displayed, if any.



If you click on the button, the form will be opened as in the figure below.

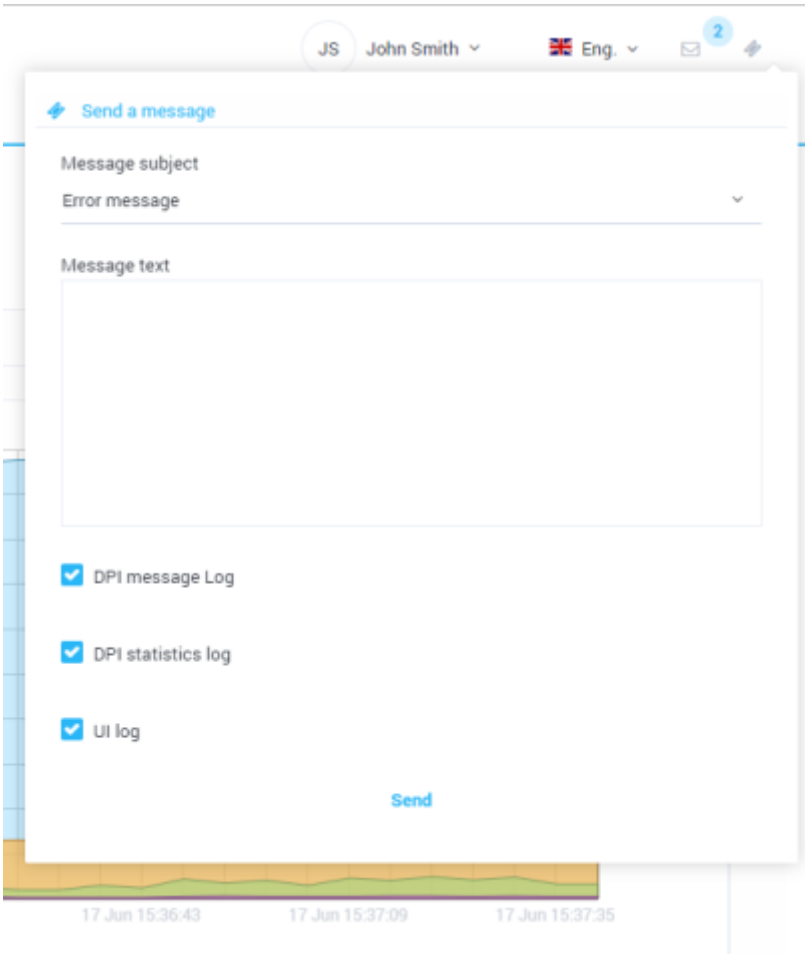


The form displays a list of recent messages, you can filter it. If you hover over a line with a message, its full text is displayed. Messages can be marked as read and deleted.

Report a bug

In the upper right corner of the program there is a Ticket button, which allows you to send a message to the technical service.

When you click on the button, the form is opened as shown in the figure below.



The form allows you to select a topic, write text, attach logs. The message is sent to the technical service.

In case of abnormal program termination the error message is displayed in the center of the screen. Also you can send a message with the error text to the technical service.

Description of the JSON-RPC software interface

Description

All requests to the server are sent by the POST method. The response headers containt Cookie. They have to be transmitted in all the subsequent requests.

An array of [] objects of following type is transferred within the request body

The example of request:

```
{
    "jsonrpc":"2.0",
    "method":"Api_GetAppVersion", -- the method name
    "params":{}, -- Object with parameters
    "id":1515659482430 - request identifier (it can be any random number)

}
```

If the request contains an array of several objects, then corresponding response will also contain an array. If the request contains an array of just one object, then corresponding response will return one object instead, not an array.

Note: the letter case and the type of parameters matters.

The response example:

```
{
    "jsonrpc":"2.0",
    "id":1515659482430, -- identifier specified in request
    "result":{
        "success":true, -- result (Boolean value)
        "data":"2.0.0" -- data (string, array, object)
    }

}
```

The response may be returned with an error. Errors are divided into two categories.

- Global - are processed at the json-rpc module level. For example incorrect parameters were passed. The error is displayed in the error field.

```
{
    "jsonrpc":"2.0",
    "id":1515660273515,
    "error":{
        "code":6000,
        "message":"Invalid parameters",
        "data":{
            "errors":[
                {
                    "code":"required_field",
                    "message":"Mandatory parameter is not passed or it
is empty",
                    "object_name":"Test"
                }
            ]
        }
```

```
    }

}
```

- Local - are processed at the API level inside the called function. For example, error occurred while writing to the database. The error is displayed in the result.error field. In this case, result.success = false.

```
{
    "jsonrpc":"2.0",
    "id":1515660902722,
    "result":{
        "success":false,
        "error":{
            "code":-203,
            "msg":"Duplicate entry"
        },
        "data":null
    }

}
```

Some example requests with CURL

Get the application version:

```
curl -b cookie.txt -c cookie.txt -X POST -k -i
'https://192.168.1.123/api/jsonrpc' --data
'[{"jsonrpc":"2.0","method":"Api_GetAppVersion","params":{},"id":15634364952
88}]'
```

Make authorization:

```
curl -b cookie.txt -c cookie.txt -X POST -k -i
'https://192.168.1.123/api/jsonrpc' --data
'[{"jsonrpc":"2.0","method":"Api_Login","params":{"username":"admin","passwo
rd":"vasexperts","remember":t
rue},"id":1563438645838}]'
```

Get DPI configuration file:

```
curl -b cookie.txt -c cookie.txt -X POST -k -i
'https://192.168.1.123/api/jsonrpc' --data
'[{"jsonrpc":"2.0","method":"Api_GetDpiConfig","params":{"Id":1},"id":156344
8913831}]'
```

Common Functions

Api_GetAppVersion

Displays the application version.

There are no parameters.

The request example:

```
[{
    "jsonrpc":"2.0",
    "method":"Api_GetAppVersion",
    "params":{},
    "id":1515659482430

}]
```

The response example:

```
{
    "jsonrpc":"2.0",
    "id":1515659482430,
    "result":{
        "success":true,
        "data":"2.0.0"
    }

}
```

Api_GetDics

Outputs dictionaries.

Parameters:

- DicsKeys - Array:list of dictionary identifiers. Displays all of them by default.

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDics",
        "params":{},
        "id":1515921551526
    }
]
```

The response example:


```
{
    "jsonrpc":"2.0",
    "id":1515921551526,
    "result":{
        "success":true,
        "data":{
            "RolesDic":{
                "1":{
                    "dic_id":"1",
                    "value":"Administrator"
                }
            },
            "HardwaresDic":{
                "1":{
                    "dic_id":"1",
                    "value":"Test bed. 34",
                    "ip":"x.x.x.x",
                    "port":"22",
                    "login":"dpisu"
                },
                "2":{
                    "dic_id":"2",
                    "value":"Operational test bed .83 !",
                    "ip":"y.y.y.y",
                    "port":"22",
                    "login":"dpisu"
                }
            }
        }
    }
}
```


Authentication and authorization functions

- Api_Auth
- Api_Login
- Api_Logout

Api_Auth

Checks and displays the result: whether the user is authorized or not.

There are no parameters.

The request example:


```
[
```

```
    {
        "jsonrpc":"2.0",
        "method":"Api_Auth",
        "params":{
        },
        "id":1515661809137
    }

]
```

The response example in case the user is not authorized:

```
{
    "jsonrpc":"2.0",
    "id":1515661809137,
    "error":{
        "code":7000,
        "message":"Unauthorized"
    }

}
```

The response example in case the user is authorized:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data":{
            "username":"admin"
        }
    }

}
```

Api_Login

Authorizes the user.

Parameters:

- username - string:the user name
- password - string:password
- remember - bool:password

The request example

```
[
```

```
    {
        "jsonrpc":"2.0",
        "method":"Api_Login",
        "params":{
            "username":"admin",
            "password":"vasexperts",
            "remember":true
        },
        "id":1515662219320
    }

]
```

The example of a successful response:

```
{
    "jsonrpc":"2.0",
    "id":1515662219320,
    "result":{
        "success":true,
        "data":{
            "username":"admin"
        }
    }

}
```

The example of a response if something's gone wrong:

```
{
    "jsonrpc":"2.0",
    "id":1515662233320,
    "result":{
        "success":false,
        "error":{
            "code":404,
            "msg":"Invalid credentials"
        },
        "data":null
    }

}
```

If the authorization is successfully completed, cookies will be are transmitted in the response headers. They must be transmitted in all subsequent requests.

Api_Logout

Deauthorizes the current user.

There are no parameters.

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_Logout",
        "params":{
        },
        "id":1515662234646
    }

]
```

User profile management

- Api_GetMyProfile
- Api_SaveMyProfile
- Api_ChangeMyPassword

Api_GetMyProfile

Displays the user profile.

There are no parameters.

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetMyProfile",
        "params":{
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": {
```

```
            "user_id": "1",
            "username": "admin",
            "name": "Strogiy S.S.",
            "email": "email@email.ru",
            "phone": "2-2-2",
            "company": "VasExpert",
            "position": "Administrator"
        }
    }

}
```

Api_SaveMyProfile

The user profile editing.

Parameters:

- MyProfile - object:the user model

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SaveMyProfile",
        "params":{
           "MyProfile":{
        "user_id": "1",
                "username": "admin",
                "name": "Strogiy S.S.",
                "email": "email@email.ru",
                "phone": "2-2-2",
                "company": "VasExpert",
                "position": "Administrator"
        }
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
```

```
        "data": {
            "user_id": "1",
            "username": "admin",
            "name": "Strogiy S.S.",
            "email": "email@email.ru",
            "phone": "2-2-2",
            "company": "VasExpert",
            "position": "Administrator"
        }
    }

}
```

Api_ChangeMyPassword

The user password changing.

Parameters:

- OldPassword - object:old password
- NewPassword - string:new password

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SaveMyProfile",
        "params":{
           "OldPassword":"vasexperts",
       "NewPassword":"vasexperts1"
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": {
            "user_id": "1",
            "username": "admin",
            "name": "Strogiy S.S.",
            "email": "email@email.ru",
```

```
            "phone": "2-2-2",
            "company": "VasExpert",
            "position": "Administrator",
            "password":
"$2y$10$t/PdsQVXr927wyunbdET1.EHtuxTBg4iKmtHlJ4jfmU0XR4qPUANu"
        }
    }

}
```

User management

- Api_GetUsers
- Api_SaveUser
- Api_DeleteUser

Api_GetUsers

Allows to get list of users.

Parameters:

- Id - int:user identifier(optional parameter)

The request example using the user identifier:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetUsers",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The example of successful response to request containing the user identifier:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": {
            "user_id": "1",
            "username": "admin",
```

```
            "name": "Strogiy S.S.",
            "email": "email@email.ru",
            "phone": "2-2-2",
            "company": "VasExpert",
            "position": "Administrator",
            "role_sections": "hardware.read,hardware.write,..."
        }
    }

}
```

The example of successful response to request without parameters:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": [
            {
                "user_id": "1",
                "username": "admin",
                "name": "Strogiy S.S.",
                "email": "email@email.ru",
                "phone": "2-2-2",
                "company": "VasExpert",
                "position": "Administrator",
                "role_sections": "hardware.read,hardware.write,..."
            },
            ...
        ]
    }

}
```

Api_SaveUser

Create/modify the user data.

Parameters:

- user - object:the user model

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SaveUser",
```

```
        "params":{
           "user":{
        "user_id": "1",
                "username": "admin",
                "name": "Strogiy S.S.",
                "email": "email@email.ru",
                "phone": "2-2-2",
                "company": "VasExpert",
                "position": "Administrator",
                "password":"vasexperts",
                "role": "1",
                "role_sections": "hardware.read,hardware.write,..."
       }
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": {
            "user_id": "1",
            "username": "admin",
            "name": "Strogiy S.S.",
            "email": "email@email.ru",
            "phone": "2-2-2",
            "company": "VasExpert",
            "position": "Administrator",
            "password":
"$2y$10$rxlWVJdRybSf9N6nAQE9j.i2LrSTbpGzoiDwsVVsAP9OQ5vDY0uhu"
        }
    }

}
```

Api_DeleteUser

Deletes the user.

Parameters:

- user - object:the user model

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_DeleteUser",
        "params":{
           "user":{
        "user_id": "2",
                "username": "admin",
                "name": "Strogiy S.S.",
                "email": "email@email.ru",
                "phone": "2-2-2",
                "company": "VasExpert",
                "position": "Administrator",
                "password":"vasexperts",
                "role": "1",
                "role_sections": "hardware.read,hardware.write,..."
       }
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Roles management

- Api_GetRoles
- Api_SaveRole
- Api_DeleteRole

Api_GetRoles

Allows to get a list of users.

Parameters:

- Id - int:role identifier(optional parameter)

The example of request with the role ID:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetRoles",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The example of successful response to request containing the role ID:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": [
            {
                "role_id": "1",
                "role": "Администратор",
                "sections": "hardware.read,hardware.write,..."
            }
        ]
    }

}
```

The example of a successful response to a request without parameters:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": [
            {
                "role_id": "1",
                "role": "Administrator",
                "sections": "hardware.read,hardware.write,..."
            },
            ...
        ]
```

```
    }

}
```

Api_SaveRole

Creating/modifying the role data.

Parameters:

- role - object:the role model

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SaveRole",
        "params":{
           "role":{
        "role_id": "1",
                "role": "Administrator",
                "sections": "hardware.read,hardware.write,..."
       }
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": {
            "role_id": "1",
            "role": "Administrator",
            "sections": "hardware.read,hardware.write,..."
        }
    }

}
```

Api_DeleteRole

Deleting a role.

Parameters:

- role - object:the role model

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_DeleteRole",
        "params":{
           "role":{
        "role_id": "1",
                "role": "Administrator",
                "sections": "hardware.read,hardware.write,..."
       }
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Device control

- Api_GetHardwares
- Api_SaveHardware
- Api_DeleteHardware
- Api_RestartHardware
- Api_ReloadHardware
- Api_TestConnectToHardware
- Api_GetDpiConfig
- Api_ValidateDpiConfig
- Api_SetDpiConfig

- Api_GetDpiConfigHistory
- Api_GetDpiConfigFile
- Api_GetDpiInfo
- Api_GetDpiResourcesUsageTick
- Api_GetDpiResourcesStatsUsageForPeriod
- Api_ProcessDpiStatLog
- Api_GetStatLogTail
- Api_DownloadStatLog
- Api_GetAlertLogTail
- Api_DownloadAlertLog

Api_GetHardwares

Get the list of equipments.

Parameters:

- Id - int:hardware identifier(optional identifier)

The request using the hardware identifier example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetHardwares",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

Example of a successful response to a request with a hardware identifier:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": [
            {
                "hardware_id": "1",
                "name": "Test bed. 34",
                "ip": "188.227.73.34",
                "port": "22",
                "login": "admin",
                "password": "vasexperts",
                "ssl_key": "",
```

```
                "sudocheck": "1",
                "load_cs": 0,
                "port_cs": "1500",
                "protocol_cs": "udp",
                "ip_cs": "217.71.228.148",
                "status_cs": 0
            }
        ]
    }

}
```

The example of successful response to the request without parameters:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": [
            {
                "hardware_id": "1",
                "name": "Test bed. 34",
                "ip": "188.227.73.34",
                "port": "22",
                "login": "admin",
                "password": "vasexperts",
                "ssl_key": "",
                "sudocheck": "1",
                "load_cs": 0,
                "port_cs": "1500",
                "protocol_cs": "udp",
                "ip_cs": "217.71.228.148",
                "status_cs": 0
            },
            ...
        ]
    }

}
```

Api_SaveHardware

Creation/modification of the equipment data.

Parameters:

- hardware - object:the equipment model

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SaveHardware",
        "params":{
           "hardware":{
        "hardware_id": "1",
                "name": "Test bed. 34",
                "ip": "188.227.73.34",
                "port": "22",
                "login": "admin",
                "password": "vasexperts",
                "ssl_key": "",
                "sudocheck": "1",
                "load_cs": 0,
                "port_cs": "1500",
                "protocol_cs": "udp",
                "ip_cs": "217.71.228.148",
                "status_cs": 0
       }
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": {
            "hardware_id": "1",
            "name": "Test bed. 34",
            "ip": "188.227.73.34",
            "port": "22",
            "login": "admin",
            "password": "vasexperts",
            "ssl_key": "",
            "sudocheck": "1",
            "load_cs": 0,
            "port_cs": "1500",
            "protocol_cs": "udp",
            "ip_cs": "217.71.228.148",
            "status_cs": 0
        }
```

```
    }

}
```

Api_DeleteHardware

Equipment removal.

Параметры:

- hardware - object:the equipment model

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_DeleteRole",
        "params":{
           "hardware":{
        "hardware_id": "1",
                "name": "Test bed. 34",
                "ip": "188.227.73.34",
                "port": "22",
                "login": "admin",
                "password": "vasexperts",
                "ssl_key": "",
                "sudocheck": "1",
                "load_cs": 0,
                "port_cs": "1500",
                "protocol_cs": "udp",
                "ip_cs": "217.71.228.148",
                "status_cs": 0
       }
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }
```

```
}
```

Api_RestartHardware

Equipment restart.

Parameters:

- Id - int:the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_RestartHardware",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Api_ReloadHardware

On-the-fly parameters updating.

Parameters:

- Id - int:the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
```

```
        "method":"Api_ReloadHardware",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Api_TestConnectToHardware

Equipment availability test.

Parameters:

- Id - int:the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_TestConnectToHardware",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
```

```
        "success":true
    }

}
```

Api_GetDpiConfig

Get the DPI equipment configuration.

Parameters:

- Id - int:the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDpiConfig",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data":"string:content"
    }

}
```

Api_ValidateDpiConfig

Validation of the DPI equipment configuration.

Parameters:

- Id - int:the equipment identifier
- Config - string: configuration

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_ValidateDpiConfig",
        "params":{
           "Id":1,
           "Config":"string:content"
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data":"FastDPI 7.4 Minsk (Dec 12 2017) : Check configuration
'/tmp/dpi/fastdpi.conf' : \n\nResult check
                configuration : SUCCESS\n"
    }

}
```

Api_SetDpiConfig

Setting DPI configuration to the equipment.

Parameters:

- Id - int:the equipment identifier
- Config - string: configuration

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SetDpiConfig",
        "params":{
           "Id":1,
           "Config":"string:content"
        },
        "id":1515661809137
```

```
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Api_GetDpiConfigHistory

Getting the history of DPI configurations on the hardware.

Parameters:

- Id - int:the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDpiConfigHistory",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": [
            "2015.03.22.03.26.12.000000.fastdpi.conf",
            "2015.03.22.03.26.22.000000.fastdpi.conf",
            "2015.03.22.19.03.53.000000.fastdpi.conf",
```

```
            "2015.03.22.19.44.35.000000.fastdpi.conf",
            "2015.03.23.01.30.11.000000.fastdpi.conf",
            "2015.03.23.01.31.26.000000.fastdpi.conf",
            ...
        ]
    }

}
```

Api_GetDpiConfigFile

Get the content of DPI configuration file.

Parameters:

- Id - int:the equipment identifier
- File - string:the file name

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDpiConfigFile",
        "params":{
           "Id":1,
       "File":"2015.03.22.03.26.12.000000.fastdpi.conf"
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": "string:content"
    }

}
```

Api_GetDpiInfo

To get the DPI equipment information.

Parameters:

- Id - int:the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDpiInfo",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": {
            "FastDPI": "FastDPI 7.4 Minsk (Dec 12 2017)",
            "Architecture": "x86_64",
            "CPU op-mode(s)": "32-bit, 64-bit",
            "Byte Order": "Little Endian",
            "CPU(s)": "8",
            "On-line CPU(s) list": "0-7",
            "Thread(s) per core": "2",
            "Core(s) per socket": "4",
            "Socket(s)": "1",
            "NUMA node(s)": "1",
            "Vendor ID": "GenuineIntel",
            "CPU family": "6",
            "Model": "58",
            "Stepping": "9",
            "CPU MHz": "3392.160",
            "BogoMIPS": "6784.32",
            "Virtualization": "VT-x",
            "L1d cache": "32K",
            "L1i cache": "32K",
            "L2 cache": "256K",
            "L3 cache": "8192K",
            "NUMA node0 CPU(s)": "0-7"
```

```
        }
    }

}
```

Api_GetDpiResourcesUsageTick

To get information about the current state of equipment performance.

Parameters:

- Id - int:the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDpiResourcesUsageTick",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": {
            "cpus_info": null,
            "cpus_usage": {
                "stat_key": 1,
                "us": 0.40000000000000002,
                "sy": 3.5,
                "ni": 0,
                "idle": 95.299999999999997,
                "wa": 0.69999999999999996,
                "hi": 0,
                "si": 0,
                "st": 0,
                "date": "2018.01.13 10:43:03"
            },
            "mem_usage": {
```

```
            "stat_key": 2,
            "total": 8030376000,
            "used": 2301548000,
            "free": 5728828000,
            "buffers": 138556000,
            "date": "2018.01.13 10:43:03"
        },
        "swap_usage": {
            "stat_key": 2,
            "total": 2097144000,
            "used": 42040000,
            "free": 2055104000,
            "cached": 795280000,
            "date": "2018.01.13 10:43:03"
        },
        "top_processes": [
            {
                "pid": "23650",
                "virt": "10.0g",
                "res": "997m",
                "shr": "10m",
                "cpu": 110.59999999999999,
                "mem": 12.699999999999999,
                "command": "fastdpi_main",
                "date": null
            },
            {
                "pid": "13347",
                "virt": "98.0m",
                "res": "3960",
                "shr": "3000",
                "cpu": 9.9000000000000004,
                "mem": 0,
                "command": "sshd",
                "date": null
            },
            {
                "pid": "13342",
                "virt": "15036",
                "res": "1380",
                "shr": "996",
                "cpu": 4,
                "mem": 0,
                "command": "top",
                "date": null
            },
            {
                "pid": "1084",
                "virt": "0",
                "res": "0",
                "shr": "0",
```

```
                    "cpu": 1,
                    "mem": 0,
                    "command": "kauditd",
                    "date": null
                },
                {
                    "pid": "13351",
                    "virt": "98.0m",
                    "res": "2004",
                    "shr": "1024",
                    "cpu": 1,
                    "mem": 0,
                    "command": "sshd",
                    "date": null
                },
                {
                    "pid": "1",
                    "virt": "19232",
                    "res": "884",
                    "shr": "748",
                    "cpu": 0,
                    "mem": 0,
                    "command": "init",
                    "date": null
                },
                {
                    "pid": "2",
                    "virt": "0",
                    "res": "0",
                    "shr": "0",
                    "cpu": 0,
                    "mem": 0,
                    "command": "kthreadd",
                    "date": null
                },
                {
                    "pid": "3",
                    "virt": "0",
                    "res": "0",
                    "shr": "0",
                    "cpu": 0,
                    "mem": 0,
                    "command": "migration/0",
                    "date": null
                },
                {
                    "pid": "4",
                    "virt": "0",
                    "res": "0",
                    "shr": "0",
                    "cpu": 0,
```

```
            "mem": 0,
            "command": "ksoftirqd/0",
            "date": null
        },
        {
            "pid": "5",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/0",
            "date": null
        },
        {
            "pid": "6",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "watchdog/0",
            "date": null
        },
        {
            "pid": "7",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/1",
            "date": null
        },
        {
            "pid": "8",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/1",
            "date": null
        },
        {
            "pid": "9",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
```

```
            "command": "ksoftirqd/1",
            "date": null
        },
        {
            "pid": "10",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "watchdog/1",
            "date": null
        },
        {
            "pid": "11",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/2",
            "date": null
        },
        {
            "pid": "12",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/2",
            "date": null
        },
        {
            "pid": "13",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "ksoftirqd/2",
            "date": null
        },
        {
            "pid": "14",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "watchdog/2",
```

```
            "date": null
        },
        {
            "pid": "15",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/3",
            "date": null
        },
        {
            "pid": "16",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/3",
            "date": null
        },
        {
            "pid": "17",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "ksoftirqd/3",
            "date": null
        },
        {
            "pid": "18",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "watchdog/3",
            "date": null
        },
        {
            "pid": "19",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/4",
            "date": null
```

```
        },
        {
            "pid": "20",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/4",
            "date": null
        },
        {
            "pid": "21",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "ksoftirqd/4",
            "date": null
        },
        {
            "pid": "22",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "watchdog/4",
            "date": null
        },
        {
            "pid": "23",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/5",
            "date": null
        },
        {
            "pid": "24",
            "virt": "0",
            "res": "0",
            "shr": "0",
            "cpu": 0,
            "mem": 0,
            "command": "migration/5",
            "date": null
        },
```

```
                {
                    "pid": "25",
                    "virt": "0",
                    "res": "0",
                    "shr": "0",
                    "cpu": 0,
                    "mem": 0,
                    "command": "ksoftirqd/5",
                    "date": null
                },
                {
                    "pid": "26",
                    "virt": "0",
                    "res": "0",
                    "shr": "0",
                    "cpu": 0,
                    "mem": 0,
                    "command": "watchdog/5",
                    "date": null
                }
            ],
            "date": "2018.01.13 10:43:03",
            "cpus_count": "8"
        }
    }

}
```

Api_GetDpiResourcesStatsUsageForPeriod

To get information about the DPI equipment performance for the period.

Параметры:

- Id - int:the equipment identifier
- Period - object:...(optional parameter)
- PeriodType - string:...(optional parameter)

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDpiResourcesStatsUsageForPeriod",
        "params":{
            "Id":1,
            "Period":{
                "start":"2018-01-13T21:00:00.000Z",
                "end":"2018-01-13T21:00:00.000Z"
            },
```

```
            "PeriodType":"H"
        },
        "id":1515921656764
    }
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515921656764,
    "result":{
        "success":true,
        "data":{
            "2018-01-14 11":{
                "cpus_usage":{
                    "stat_key":"1",
                    "us":"0.12121212121212127",
                    "sy":"0.47272727272727294",
                    "ni":"0",
                    "idle":"99.14545454545457",
                    "wa":"0.2787878787878788",
                    "hi":"0",
                    "si":"0",
                    "st":"0",
                    "date":"2018-01-14 11"
                },
                "mem_usage":{
                    "stat_key":"2",
                    "virt":"10787827712",
                    "res":"3747196928",
                    "date":"2018-01-14 11"
                },
                "dnas_usage":{
                    "1_0_out":{
                        "stat_key":"1_0_out",
                        "cluster_id":"1",
                        "dna_id":"0",
                        "act_rcvd_pkts":"165.87878787878788",
                        "act_send_pkts":"0",
                        "act_esnd_pkts_err":"0",
                        "act_drop_pkts":"0",
                        "act_eemt_pkts_err":"0",
                        "date":"2018-01-14 11"
                    },
                    "1_1_in":{
                        "stat_key":"1_1_in",
                        "cluster_id":"1",
                        "dna_id":"1",
                        "act_rcvd_pkts":"0",
                        "act_send_pkts":"165.87878787878788",
```

```
                    "act_esnd_pkts_err":"0",
                    "act_drop_pkts":"0",
                    "act_eemt_pkts_err":"0",
                    "date":"2018-01-14 11"
                }
            },
            "ifs_usage":{
                "1":{
                    "cluster_id":"1",
                    "abs_rcvd_pkts":"1755914.8181818181",
                    "abs_rcvd_bytes":"1572754593.4545455",
                    "abs_rcvd_pkts_dropped":"0",
                    "act_captured_pkts_sec":"0.03606060606060608",
                    "act_processed_pkts_sec":"0.03606060606060608",
                    "act_send_pkts_sec":"0.03606060606060608",
                    "date":"2018-01-14 11"
                }
            },
            "http_usage":{
                "stat_key":"5",
                "url":"1517",
                "url_lock":"0",
                "ssl":"2653",
                "ssl_lock":"0",
                "cna":"8",
                "cna_lock":"0",
                "sni":"2645",
                "sni_lock":"0",
                "quic":"392",
                "quic_lock":"0",
                "chnprc":"0",
                "ccheck":"487579",
                "ccheck_ip_check":null,
                "ccheck_lock":null,
                "ftp":0,
                "ftp_lock":0,
                "smtp":0,
                "smtp_lock":0,
                "pop3":0,
                "pop3_lock":0,
                "imap":0,
                "imap_lock":0,
                "xmpp":0,
                "xmpp_lock":0,
                "date":"2018-01-14 11"
            },
            "date":"2018-01-14 11"
        },
        "2018-01-14 12":{
            "cpus_usage":{
                "stat_key":"1",
```

```
            "us":"2.4529953917050675",
            "sy":"1.2198156682027643",
            "ni":"0",
            "idle":"95.97419354838706",
            "wa":"0.3612903225806449",
            "hi":"0",
            "si":"0",
            "st":"0",
            "date":"2018-01-14 12"
        },
        "mem_usage":{
            "stat_key":"2",
            "virt":"10787827712",
            "res":"3747747245.419355",
            "date":"2018-01-14 12"
        },
        "dnas_usage":{
            "1_1_in":{
                "stat_key":"1_1_in",
                "cluster_id":"1",
                "dna_id":"1",
                "act_rcvd_pkts":"23789.79262672811",
                "act_send_pkts":"175827.29493087556",
                "act_esnd_pkts_err":"0",
                "act_drop_pkts":"0",
                "act_eemt_pkts_err":"0",
                "date":"2018-01-14 12"
            },
            "1_0_out":{
                "stat_key":"1_0_out",
                "cluster_id":"1",
                "dna_id":"0",
                "act_rcvd_pkts":"175827.29493087556",
                "act_send_pkts":"23789.79262672811",
                "act_esnd_pkts_err":"0",
                "act_drop_pkts":"0",
                "act_eemt_pkts_err":"0",
                "date":"2018-01-14 12"
            }
        },
        "ifs_usage":{
            "1":{
                "cluster_id":"1",
                "abs_rcvd_pkts":"1795976.133640553",
                "abs_rcvd_bytes":"1597939948.7695854",
                "abs_rcvd_pkts_dropped":"0",
                "act_captured_pkts_sec":"22.814608294930885",
                "act_processed_pkts_sec":"22.814608294930885",
                "act_send_pkts_sec":"22.814608294930885",
                "date":"2018-01-14 12"
            }
```

```
                },
                "http_usage":{
                    "stat_key":"5",
                    "url":"1527.6036866359448",
                    "url_lock":"0",
                    "ssl":"2943.2258064516127",
                    "ssl_lock":"0",
                    "cna":"8",
                    "cna_lock":"0",
                    "sni":"2935.2258064516127",
                    "sni_lock":"0",
                    "quic":"392",
                    "quic_lock":"0",
                    "chnprc":"0",
                    "ccheck":"488168.02764976956",
                    "ccheck_ip_check":null,
                    "ccheck_lock":null,
                    "ftp":0,
                    "ftp_lock":0,
                    "smtp":0,
                    "smtp_lock":0,
                    "pop3":0,
                    "pop3_lock":0,
                    "imap":0,
                    "imap_lock":0,
                    "xmpp":0,
                    "xmpp_lock":0,
                    "date":"2018-01-14 12"
                },
                "date":"2018-01-14 12"
            }
        }
    }

}
```


Api_ProcessDpiStatLog

Starts the DPI statistics log processing. It is usually started automatically on a schedule.

Parameters:

- Id - int:the equipment identifier

The request example:


```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_ProcessDpiStatLog",
```

```
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Api_GetStatLogTail

Gets the "tail" of the DPI statistics log.

Parameters:

- Id - int:the equipment identifier
- Tail - int: the "tail" length

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetStatLogTail",
        "params":{
           "Id":1,
       "Tail":20
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
```

```
    "result":{
        "success":true,
        "data": "\ttotal : allocate=165/7555650\n[STAT
][2018/01/13-13:56:08:375363] Detailed statistics on HTTP
                :\n\tthread_slave=0 :\n\t\turl/lock=4/0 ( 5,0,0 )( 1,1,0
)\n\t\tssl/lock=25/0 ( 0,1,0 )( 1,2,1
                )\n\t\t\tcna/lock=0/0 ( 0,1 )\n\t\t\tsni/lock=25/0 ( 0,0
)\n\t\tquic/lock=0/0 ( 0,0,0 )( 0,0,0
                )\n\t\tchnprc=0\n\t\tccheck/ip_check/lock=309/20/0\n\tTotal
: \n\t\turl/lock=4/0 ( 5,0,0 )(
                1,1,0,98879 )\n\t\tssl/lock=25/0 ( 0,1,0 )( 1,2,1,196647
)\n\t\t\tcna/lock=0/0 ( 0,1
                )\n\t\t\tsni/lock=25/0 ( 0,0 )\n\t\tquic/lock=0/0 ( 0,0,0
)( 0,0,0,0
                )\n\t\tchnprc=0\n\t\tccheck/ip_check/lock=309/20/0\n[STAT
][2018/01/13-13:56:08:375374] [BRAS]
                ARP statistics:\n\tprocessed: subs->inet=0, inet->subs=0;
originated=0\n"
    }

}
```

Api_DownloadStatLog

Gets the statistics log file identifier to be used for downloading.

Parameters:

- Id - int:the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_DownloadStatLog",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
```

```
    "result":{
        "success":true,
        "data": "fq4KpGuICfAxICtX5lAC"
    }

}
```

When the identifier is received, the file can be obtained from the link

```
http://<server ip address>/api/file/fq4KpGuICfAxICtX5lAC
```

Api_GetAlertLogTail

Gets "tail" of the DPI messages log.

Parameters:

- Id - int:the equipment identifier
- Tail - int: the "tail" length

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetAlertLogTail",
        "params":{
           "Id":1,
       "Tail":20
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": "\t\tbras_max_session_duration  =604800
seconds\n\t\tbras_dhcp_request_delay    =300 seconds\n\tARP
                proxy:\n\t\tbras_arp_proxy             =0x0000 (local:off,
gateway:off)\n\t\tbras_arp_ip
                =0.0.0.0 (not specified)\n\t\tbras_arp_mac
=00:00:00:00:00:00\n\tBRAS
                services:\n\t\tbras_dhcp_auth_mix       =1
```

```
(enabled)\n\t\tbras_ip_source_guard     =0
                  (disabled)\n\t\tbras_terminate_local     =0
(disabled)\n\t\tbras_vlan_terminate      =0
                  (disabled)\n\t\tbras_vlan_subst
=\n\t\tbras_terminate_l2        =0
                  (disabled)\n\t\tbras_term_by_as          =0
(disabled)\n\t\tbras_gateway_ip
                  =0.0.0.0\n\t\tbras_gateway_mac
=00:00:00:00:00:00\n[COMMON  ][2018/01/13-13:32:48:364905]
                  [0x7fac68409820] BRAS PPPoE: disabled\n[INFO
][2018/01/13-13:32:48:365180][0x7fac68409820] DSCP
                  settings is loaded.\n[WARNING
][2018/01/13-13:35:58:302925][0x7faa21306700] NFLW : very long
                  operation to send data, duration=11000669 msec,
tmout=10000380 msec, cntr=1\n[ERROR   ]
                  [2018/01/13-13:37:13:630075][0x7faa138fe700] bpm : Error
reset_bypass_wd_timer, if='dna0',
                  errno=95 : Operation not supported\n"
    }

}
```

Api_DownloadAlertLog

To get the messages log file identifier to be used for downloading.

Parameters:

- Id - int:the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDpiInfo",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
```

```
    "result":{
        "success":true,
        "data": "tjFNBnsRkQmXlJbBsCHM"
    }

}
```

When the identifier is received the file can be obtained from the link like following

```
http://<ip адрес сервера>/api/file/fq4KpGuICfAxICtX5lAC
```

Notifications management

- Api_GetLastNotifications
- Api_GetUnreadedNotificationsCount
- Api_SetNotificationsReaded
- Api_DeleteNotifications

Api_GetLastNotifications

To get the list of the latest notifacations.

Parameters:

- Page - int:the page number
- Size - int:the number of notifications in the response

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetLastNotifications",
        "params":{
           "Page":1,
       "Size":3
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
```

```
        "success":true,
        "data": {
            "page": 1,
            "size": 3,
            "count": 200,
            "rows": [
                {
                    "notif_id": "391439",
                    "hardware_hardware_id": "2",
                    "users_user_id": null,
                    "date": "2018-01-12T09:22:32",
                    "from": "Result 'check license', err_code=1",
                    "text": " Result 'check license', err_code=1 : Success:
loaded",
                    "importance": "COMMON",
                    "readed": "0",
                    "deleted": "0"
                },
                {
                    "notif_id": "391438",
                    "hardware_hardware_id": "2",
                    "users_user_id": null,
                    "date": "2018-01-12T09:22:32",
                    "from": "Check license",
                    "text": " Check license : devices dna0 <--> dna1 ...",
                    "importance": "COMMON",
                    "readed": "0",
                    "deleted": "0"
                },
                {
                    "notif_id": "391437",
                    "hardware_hardware_id": "2",
                    "users_user_id": null,
                    "date": "2018-01-12T09:20:32",
                    "from": "bl_updater_thread",
                    "text": " bl_updater_thread :
Notifications(updatenotifications) list loaded with result, rc=0
                             : Success: loaded.",
                    "importance": "INFO",
                    "readed": "0",
                    "deleted": "0"
                }
            ]
        }
        }
    }

}
```

Api_GetUnreadedNotificationsCount

To get the number of unread CRITICAL notifications.

There are no parameters.

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetUnreadedNotificationsCount",
        "params":{
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": 123
    }

}
```

Api_SetNotificationsReaded

Marks notifications as read.

Parameters:

- Notifs - array:the array of notifications

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SetNotificationsReaded",
        "params":{
           "Notifs":[
        {
```

```
                    "notif_id": "391439",
                    "hardware_hardware_id": "2",
                    "users_user_id": null,
                    "date": "2018-01-12T09:22:32",
                    "from": "Result 'check license', err_code=1",
                    "text": " Result 'check license', err_code=1 : Success:
loaded",
                    "importance": "COMMON",
                    "readed": "0",
                    "deleted": "0"
                },
                ...
            ]
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Api_DeleteNotifications

Deletes notifications.

Parameters:

- Notifs - array:the array of notifications

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_DeleteNotifications",
        "params":{
           "Notifs":[
        {
                    "notif_id": "391439",
                    "hardware_hardware_id": "2",
```

```
                    "users_user_id": null,
                    "date": "2018-01-12T09:22:32",
                    "from": "Result 'check license', err_code=1",
                    "text": " Result 'check license', err_code=1 : Success:
loaded",
                    "importance": "COMMON",
                    "readed": "0",
                    "deleted": "0"
                },
                ...
            ]
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Errors sending

- Api_SendError
- Api_LogError

Api_SendError

Sends an error message to the mail specified in the configuration.

Parameters:

- Params - object: the error message model

Sending configurations:

- subject - string: message subject
- body - string: message body
- hardware_id - int: equipment identifier
- alertLog - bool: to include (or exclude) notification logs file in a message
- statLog - bool: to include (or exclude) equipment status logs file in a message

- uiLog - bool: to include (or exclude) dpiui logs file in a message

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SendError",
        "params":{
           "Params":{
              "hardware_id": "1",
              "subject":"test",
              "body":"test",
              "alertLog":true,
              "statLog":true,
              "uiLog":true
           }
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Api_LogError

Writes the error information to the dpiui log file.

Parameters:

- Content - string: error message body

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_LogError",
        "params":{
```

```
            "Content":"error text"
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Subscriber management

- Api_GetSubscribers
- Api_SaveSubscriber
- Api_DeleteSubscriber

Api_GetSubscribers

Gets the list of subscribers.

Parameters:

- Params - object: search options (optional parameter)

Possible search parameters Params:

- page - int: page number
- size - int: dimension of array containing the result
- hardware_id - int: equipment identifier
- login - string: user name
- login_equal - string:...
- ip - string: user ip address
- ip_equal - string:...
- bind_type - ...
- services - ...
- order_by - string: field used for ordering the subscribers list
- order_dir - string: ordering the list in ascending order ("asc") / descending ("desc")

The request example:

```
[

    {
        "jsonrpc":"2.0",
        "method":"Api_GetSubscribers",
        "params":{
            "Params":{
                "hardware_id":"1",
                "page":0,
                "size":100,
                "login":"",
                "ip":"",
                "bind_type":"",
                "services":""
            }
        },
        "id":1515929803508
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515929803508,
    "result":{
        "success":true,
        "data":{
            "page":0,
            "size":100,
            "count":6,
            "rows":[
                {
                    "hardware_id":"1",
                    "login":"llllyyy",
                    "ip":"5.5.5.7",
                    "bind_type":"1",
                    "services":"5,"
                },
                {
                    "hardware_id":"1",
                    "login":"user11",
                    "ip":"1.1.1.1",
                    "bind_type":"0",
                    "services":"4,"
                },
                {
                    "hardware_id":"1",
                    "login":"user22",
```

```
                    "ip":"1.1.1.3",
                    "bind_type":"1",
                    "services":"2,3,4,5,6,"
                },
                {
                    "hardware_id":"1",
                    "login":"user3",
                    "ip":"2.2.2.2",
                    "bind_type":"1",
                    "services":"2,3,4,5,6,"
                },
                {
                    "hardware_id":"1",
                    "login":"user4",
                    "ip":"3.3.3.3",
                    "bind_type":"1",
                    "services":"4,"
                },
                {
                    "hardware_id":"1",
                    "login":"userxx",
                    "ip":"5.5.5.6",
                    "bind_type":"1",
                    "services":"3,4,6,"
                }
            ]
        }
    }

}
```

Api_SaveSubscriber

Saves the subscriber model.

Parameters:

- Id - int: equipment identifier
- Subscriber - object: subscriber model containing new data
- SubscriberOld - object: subscriber model containing old data. If this parameter is specified, the subscriber data and its activated services are edited. Otherwise (if not specified) a new subscriber will be created.

The example of request when creating a subscriber:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SaveSubscriber",
        "params":{
```

```
            "Id":1,
            "Subscriber":{
                "id":1515929803535,
                "hardware_id":0,
                "login":"test",
                "ip":"77.77.77.77",
                "bind_type":"0",
                "services":"4,6,",
                "tariff":"",
                "valid":true,
                "oldSubscriber":{
                    "id":1515929803535,
                    "hardware_id":0,
                    "login":"",
                    "ip":"",
                    "bind_type":1,
                    "services":"",
                    "tariff":"",
                    "valid":true
                }
            },
            "SubscriberOld":{
                "id":1515929803535,
                "hardware_id":0,
                "login":"",
                "ip":"",
                "bind_type":1,
                "services":"",
                "tariff":"",
                "valid":true
            }
        },
        "id":1515929803535
    }
]
```


The successful response example when creating a subscriber:


```
{

    "jsonrpc":"2.0",
    "id":1515929803535,
    "result":{
        "success":true,
        "data":{
            "bind":{
                "success":true,
                "data":{
                    "bind_result":{
                        "total":"1",
                        "success":"1",
```

```
                        "fail":"0",
                        "were_set_before":"0",
                        "were_not_set_before":"0",
                        "data":{
                        }
                    },
                    "unbind_result":null
                }
            },
            "services":{
                "success":true,
                "data":{
                    "bind_result":{
                        "total":"1",
                        "success":"1",
                        "fail":"0",
                        "were_set_before":"0",
                        "were_not_set_before":"0",
                        "data":{
                        }
                    },
                    "unbind_result":null
                }
            },
            "sync_bind":{
                "success":true,
                "data":0.0988309383392334
            },
            "sync_services":{
                "success":true,
                "data":0.08402705192565918
            }
        }
    }

}
```

The request example when editing a subscriber:

```
[

    {
        "jsonrpc":"2.0",
        "method":"Api_SaveSubscriber",
        "params":{
            "Id":1,
            "Subscriber":{
                "id":1515929804490,
                "hardware_id":"1",
                "login":"test123",
                "ip":"77.77.77.77",
```

```
                "bind_type":"0",
                "services":"1,4,6,",
                "tariff":"",
                "valid":true,
                "oldSubscriber":{
                    "id":1515929804490,
                    "hardware_id":"1",
                    "login":"test",
                    "ip":"77.77.77.77",
                    "bind_type":"0",
                    "services":"4,6,",
                    "tariff":"",
                    "valid":true
                }
            },
            "SubscriberOld":{
                "id":1515929804490,
                "hardware_id":"1",
                "login":"test",
                "ip":"77.77.77.77",
                "bind_type":"0",
                "services":"4,6,",
                "tariff":"",
                "valid":true
            }
        },
        "id":1515929804490
    }

]
```

The successful response example when editing a subscriber:

```
{

    "jsonrpc":"2.0",
    "id":1515929804490,
    "result":{
        "success":true,
        "data":{
            "bind":{
                "success":true,
                "data":{
                    "bind_result":{
                        "total":"1",
                        "success":"1",
                        "fail":"0",
                        "were_set_before":"0",
                        "were_not_set_before":"0",
                        "data":{
                        }
```

```
                    },
                    "unbind_result":null
                }
            },
            "services":{
                "success":true,
                "data":{
                    "bind_result":{
                        "total":"1",
                        "success":"1",
                        "fail":"0",
                        "were_set_before":"0",
                        "were_not_set_before":"0",
                        "data":{
                        }
                    },
                    "unbind_result":{
                        "total":"1",
                        "success":"1",
                        "fail":"0",
                        "were_set_before":"0",
                        "were_not_set_before":"0",
                        "data":{
                        }
                    }
                }
            },
            "sync_bind":{
                "success":true,
                "data":0.1651449203491211
            },
            "sync_services":{
                "success":true,
                "data":0.1676170825958252
            }
        }
    }

}
```


Api_DeleteSubscriber

Deletes a subscriber.

Parameters:

- Id - int: the equipment identifier
- Subscriber - object: the subscriber model

The request example:

```
[

    {
        "jsonrpc":"2.0",
        "method":"Api_DeleteSubscriber",
        "params":{
            "Id":1,
            "Subscriber":{
                "id":1515929804638,
                "hardware_id":"1",
                "login":"test123",
                "ip":"77.77.77.77",
                "bind_type":"0",
                "services":"1,4,6,",
                "tariff":"",
                "valid":true
            }
        },
        "id":1515929804638
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515929804638,
    "result":{
        "success":true,
        "data":{
            "services":{
                "success":true,
                "data":{
                    "bind_result":null,
                    "unbind_result":{
                        "total":"1",
                        "success":"1",
                        "fail":"0",
                        "were_set_before":"0",
                        "were_not_set_before":"0",
                        "data":{
                        }
                    }
                }
            },
            "unbind":{
                "success":true,
                "data":{
                    "total":"1",
```

```
                    "success":"1",
                    "fail":"0",
                    "were_set_before":"0",
                    "were_not_set_before":"0",
                    "data":{
                    }
                }
            },
            "sync_bind":{
                "success":true,
                "data":0.07761001586914062
            },
            "sync_services":{
                "success":true,
                "data":0.0842139720916748
            }
        }
    }

}
```

Services management

Is under development

Tariff management

Is under development

Protocol prioritization management (DSCP)

- Api_GetDscpProtocolsConfig
- Api_SetDscpProtocolsConfig
- Api_GetDscpProtocolsConfigHistory
- Api_GetDscpProtocolsConfigFile

Api_GetDscpProtocolsConfig

Gets the DSCP prioritization configuration.

Parameters:

- Id - int: the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDscpProtocolsConfig",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": "ftp cs4\nhttps cs1\nBittorrent cs7\nTCP Unknown
cs7\ndefault keep\n"
    }

}
```

Api_SetDscpProtocolsConfig

Sets the DSCP protocol prioritization to be used.

Parameters:

- Id - int: the equipment identifier
- Config - string: configuration content

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SetDscpProtocolsConfig",
        "params":{
           "Id":1,
           "Config":"ftp cs4\nhttps cs1\nBittorrent cs7\nTCP Unknown
cs7\ndefault keep\n"
        },
        "id":1515661809137
    }
```

```
]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true
    }

}
```

Api_GetDscpProtocolsConfigHistory

Gets the history of DSCP prioritization configuration changes.

Parameters:

- Id - int: the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDscpProtocolsConfigHistory",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data":  [
            "2015.06.03.03.54.25.000000.protocols.dscp",
            "2015.06.03.04.00.28.000000.protocols.dscp",
            "2015.06.04.15.48.34.000000.protocols.dscp",
            ...
        ]
```

```
    }

}
```

Api_GetDscpProtocolsConfigFile

Получение файла конфигурации из истории по имени файла.

Parameters:

- Id - int: the equipment identifier
- File - string: the file name

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetDscpProtocolsConfigFile",
        "params":{
           "Id":1,
           "File":"2015.06.03.03.54.25.000000.protocols.dscp"
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": "dns cs0\nhttp cs1\nhttps cs1\nBittorrent cs7\nICMPv6
cs0\nICMP cs0\ndefault cs2\n"
    }

}
```

ASN prioritization management

- Api_GetAsnDscpConfig
- Api_SetAsnDscpConfig
- Api_GetAsnDscpConfigHistory
- Api_GetAsnDscpConfigFile

Api_GetAsnDscpConfig

Gets the ASN prioritization configuration.

Parameters:

- Id - int: the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetAsnDscpConfig",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": {
            "asn": {
                "success": true,
                "data": "192.168.1.16/32 64512\n192.168.1.111/32 64512\n"
            },
            "asn_dscp": {
                "success": true,
                "data": "64512 pass\n64512 cs0\n64513 drop\n64514
pass\n64514 peer\n"
            }
        }
    }

}
```

Api_SetAsnDscpConfig

Sets the ASN prioritization configuration.

Parameters:

- Id - int: the equipment identifier
- AsnConfig - string: ASN configuration content
- AsnDscpConfig - string: ASNDSCP configuration content

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_SetDscpProtocolsConfig",
        "params":{
           "Id":1,
           "AsnConfig":"192.168.1.16/32 64512\n192.168.1.111/32 64512\n",
       "AsnDscpConfig":"64512 pass\n64512 cs0\n64513 drop\n64514 pass\n64514
peer\n"
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": {
            "asn": {
                "success": true
            },
            "asn_dscp": {
                "success": true
            }
        }
    }

}
```

Api_GetAsnDscpConfigHistory

Gets the history of ASN prioritization configuration changes.

Parameters:

- Id - int: the equipment identifier

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetAsnDscpConfigHistory",
        "params":{
           "Id":1
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data":  [
            "2015.06.07.08.29.29.000000.asnum.dscp",
            "2015.06.07.08.30.12.000000.asnum.dscp",
            "2015.06.07.08.33.40.000000.asnum.dscp",
            ...
        ]
    }

}
```

Api_GetAsnDscpConfigFile

Gets the configuration file by file name from the history.

Parameters:

- Id - int: the equipment identifier
- File - string: the file name

The request example:

```
[
    {
        "jsonrpc":"2.0",
        "method":"Api_GetAsnDscpConfigFile",
        "params":{
           "Id":1,
           "File":"2015.06.07.08.29.29.000000.asnum.dscp"
```

```
        },
        "id":1515661809137
    }

]
```

The successful response example:

```
{

    "jsonrpc":"2.0",
    "id":1515662165576,
    "result":{
        "success":true,
        "data": "11111 pass\n11112 pass\n11113 pass\n11114 pass\n11115
pass\n11116 pass\n11117 pass\n11118
                pass\n11119 pass\n11120 pass\n64514 pass\n"
    }

}
```