

# **AKIPS**

## **API reference guide**

© 2022 AKIPS Holdings Pty Ltd

All rights reserved worldwide. No part of this document may be reproduced by any means, nor modified, decompiled, dissembled, published or distributed, in whole or in part, or translated to any electronic medium or other means, without the written consent of AKIPS Holdings Pty Ltd.

All rights, title and interest in and to the software documentation are and shall remain the exclusive property of AKIPS and its licensors.

All other trademarks contained in this document are the property of their respective owners.

## **Disclaimer**

While the publisher (AKIPS Pty Ltd) has taken every precaution in the preparation of this guide to ensure that the information and instructions contained herein are accurate at the date of publication, it makes no expressed or implied warranty of any kind, and disclaims all responsibility for errors or omissions. The publisher assumes no liability for incidental or consequential losses or damages in connection with, or arising out of, the use of the information contained herein.

<b>Edition</b>	<b>Software release</b>	<b>Date</b>
17	22.10	December 2022

# Table of Contents

<b>1</b>	<b>ABOUT THIS GUIDE .....</b>	<b>6</b>
1.1	TEXT CONVENTIONS .....	6
1.2	SYNTAX.....	7
<b>2</b>	<b>AKIPS API.....</b>	<b>8</b>
2.1	TEST ENVIRONMENT .....	8
2.2	COMMAND CONSOLE .....	8
2.3	REGEX .....	9
2.4	TIME FILTERS .....	10
2.4.1	<i>tf pairs and tf span</i> .....	13
2.4.2	<i>tf check</i> .....	15
2.5	IP ADDRESS FILTERS.....	15
<b>3</b>	<b>DATABASE OVERVIEW.....</b>	<b>18</b>
3.1	ENTITIES .....	18
3.1.1	<i>Level 1: parent</i> .....	19
3.1.2	<i>Level 2: child</i> .....	19
3.1.3	<i>Level 3: attribute</i> .....	20
3.1.4	<i>Virtual attributes</i> .....	20
3.2	GROUPS.....	21
3.2.1	<i>Super groups</i> .....	21
3.2.2	<i>Configuring groups</i> .....	23
3.3	EVENTS RECORDS.....	27
3.4	TIME SERIES.....	28
<b>4</b>	<b>CONFIG AND EVENTS .....</b>	<b>29</b>
4.1	ENTITY COMMANDS .....	29
4.1.1	<i>add</i> .....	29
4.1.2	<i>delete</i> .....	30
4.1.3	<i>get</i> .....	31
4.1.4	<i>mdelete</i> .....	32
4.1.5	<i>mget</i> .....	32
4.1.6	<i>mlist</i> .....	34
4.1.7	<i>mtype</i> .....	34
4.1.8	<i>rename</i> .....	35
4.1.9	<i>set</i> .....	35
4.2	GROUP COMMANDS.....	36
4.2.1	<i>add</i> .....	36
4.2.2	<i>assign</i> .....	37
4.2.3	<i>clear</i> .....	39
4.2.4	<i>count</i> .....	40
4.2.5	<i>delete</i> .....	40
4.2.6	<i>get</i> .....	41
4.2.7	<i>list</i> .....	41
4.2.8	<i>mget</i> .....	42
4.2.9	<i>mlist</i> .....	42
4.2.10	<i>mtype</i> .....	43
4.2.11	<i>prune</i> .....	43

4.2.12	<i>rename</i> .....	43
4.3	EVENT COMMANDS.....	44
4.3.1	<i>add</i> .....	44
4.3.2	<i>clear</i> .....	44
4.3.3	<i>delete</i> .....	45
4.3.4	<i>mdelete</i> .....	45
4.3.5	<i>mget</i> .....	45
4.3.6	<i>set</i> .....	47
4.3.7	<i>tget</i> .....	47
4.4	TIME-SERIES COMMANDS.....	48
4.4.1	<i>calc</i> .....	48
4.4.2	<i>mcalc</i> .....	48
4.4.3	<i>mget</i> .....	49
4.4.4	<i>series</i> .....	49
4.4.5	<i>top</i> .....	50
4.5	WEB API.....	50
4.5.1	<i>cURL</i> .....	52
4.5.2	<i>data</i> .....	53
<b>5</b>	<b>SYSLOG AND TRAPS REPORTER.....</b>	<b>54</b>
5.1	CSV OUTPUT.....	54
5.2	WEB API.....	56
<b>6</b>	<b>AVAILABILITY.....</b>	<b>58</b>
6.1	GROUP.....	58
6.2	DEVICE.....	58
6.3	EVENTS.....	59
<b>7</b>	<b>NETFLOW REPORTER.....</b>	<b>60</b>
7.1	CSV OUTPUT.....	62
7.2	WEB API.....	63
<b>8</b>	<b>NETFLOW TIME SERIES.....</b>	<b>65</b>
8.1	CSV OUTPUT.....	65
8.2	WEB API.....	66
<b>9</b>	<b>SWITCH PORT MAPPER.....</b>	<b>68</b>
9.1	CSV OUTPUT.....	68
9.2	WEB API.....	69
<b>10</b>	<b>UNUSED INTERFACES.....</b>	<b>70</b>
10.1	CSV OUTPUT.....	70
10.2	WEB API.....	70
<b>11</b>	<b>TCP SOCKET.....</b>	<b>72</b>
<b>12</b>	<b>PERL MODULES.....</b>	<b>73</b>
12.1	COMMON.....	73
12.1.1	<i>Useful arrays</i> .....	73
12.1.2	<i>PRINT_LINE</i> .....	74
12.1.3	<i>trim</i> .....	74
12.1.4	<i>errlog</i> .....	74
12.1.5	<i>get_localtime()</i> .....	75

12.1.6	<i>mail()</i> .....	76
12.1.7	<i>syslog()</i> .....	76
12.1.8	<i>http_send()</i> and <i>http_result()</i> .....	77
12.2	AKIPS DATABASE .....	78
12.2.1	<i>adb_send()</i> .....	78
12.2.2	<i>adb_flush()</i> .....	78
12.2.3	<i>adb_result()</i> .....	78
12.3	DISCOVER .....	78
12.3.1	<i>discover_scan()</i> .....	78
12.3.2	<i>discover_config()</i> .....	79
12.3.3	<i>discover_device_rewalk()</i> .....	79
<b>13</b>	<b>SITE SCRIPTS .....</b>	<b>80</b>
13.1	USING SITE SCRIPTS .....	80
13.2	NAMING A SITE SCRIPT.....	82
13.3	ALERTING SITE SCRIPTS .....	82

# 1 About this guide

The AKIPS *API reference guide* assists users of API code in AKIPS Network Monitoring Software.

The following **Abbreviations** (see 1.1), **Text conventions** (see 1.2) and **Syntax** (see 1.3) are used throughout AKIPS' guides.

## 1.1 Text conventions

Menu options are in **bold**.

E.g. Go to Admin > System > System Settings

**Bold** is also used for emphasis or clarity.

E.g. The **backup server** must have double the disk space of the **production server**.

Links to other parts of this guide are shown as red boxes.

E.g. The following **Abbreviations** (see 1.1), **Text conventions** (see 1.2) and **Syntax** (see 1.3) are used throughout AKIPS' guides.

Websites and email addresses are in blue.

E.g. <https://www.akips.com>

Code is in monospace.

Further:

Command syntax is in red monospace.

E.g. {ddd} {hh:mm} to {hh:mm}

Input (by the user) is in blue monospace.

E.g. `tf dump last7d`

Output (by AKIPS) is in cyan monospace.

E.g. `cisco-74-1-1 sys ip4addr = 10.74.1.1`

## 1.2 Syntax

Syntax may be presented in this guide across multiple lines due to layout constraints. When using AKIPS, you will need to run commands in a single line.

Parameters (fields expecting a substituted value) are contained within { } (braces).

E.g. `{type} {value}`

Optional parameters are contained within [ ] (square brackets).

E.g. `[index,{description}]`

Optional parameters may be nested.

E.g.  
`mlist {type} [{parent regex} [{child regex} [{attribute regex}]]]`

For values separated by a | (pipe), choose one option only.

E.g. `[any|all|not group {group name} ...]`

Multiple parameters will have an ... (ellipsis).

E.g. `not group {group name} ...`

## 2 AKIPS API

### 2.1 Test environment

AKIPS recommends that you test all API code in a test environment.

Email [keys@akips.com](mailto:keys@akips.com) to request a licence.

Refer to the 'AKIPS licence' chapter in the *AKIPS Install & upgrade guide*.

To view the video *AKIPS licence*, visit <https://vimeo.com/manage/videos/514080623>

Before you execute code in your production environment, ensure that you have a current backup.

Refer to the *AKIPS Backup & restore guide*.

### 2.2 Command console

**Warning: for expert use only.**

Only admin users may access the command console.

#### **To use the command console:**

Go to Admin > API > Command Console.

To run commands:

In the text field, enter your command/s.

Click **Run Commands**

To view your command history:

Click **History**.



## 2.3 Regex

Regex syntax depends on its underlying language (C, Perl, Java, JavaScript, Python, etc), which is responsible for compiling and executing the expression.

AKIPS is written in C and Perl, with a small amount of JavaScript.

All of our C programs are compatible with the PCRE library.

AKIPS API requires that:

- all regex strings be enclosed within `/ /` (forward slashes)
- regex that contains spaces also be enclosed within `" "` (double quotation marks).

To negate regex, use the following syntax:

```
/{regex}/
```

Be precise with your syntax so that it performs quickly and efficiently.

E.g. `/^NewYork-Router/` is faster than `/NewYork-Router/` because it specifies that the start of the match *must* begin with NewYork-Router.

For more information on regex, refer to *Regular Expressions Cookbook* (Goyvaerts & Levithan, 2012).

## 2.4 Time filters

AKIPS uses syntax for time filters in:

- reports and graphs
- threshold, alerting and availability rules.

AKIPS defines a week as 00:00:00 on Sunday to 23:59:59 on Saturday.

Syntax	Description	Example
<code>m</code>	minute	<code>7m</code>
<code>h</code>	hour	<code>2h</code>
<code>d</code>	day	<code>5d</code>
<code>w</code>	week	<code>6w</code>
<code>M</code>	month	<code>3M</code>
<code>y / YYYY</code>	year	<code>1y / 2021</code>
<code>YYYY q[1-4]</code>	year and quarter	<code>2021 q2</code>
<code>YYYY-MM</code>	year and month	<code>2021-04</code>
<code>YYYY-MM-dd</code>	year, month and day	<code>2021-04-17</code>
<code>YYYY-MM-dd hh:mm</code>	year, month, day and time	<code>2021-04-17 15:30</code>

Use the following syntax:

```
{positive integer}{m h d w M y}
{ddd} {hh:mm} to {hh:mm}
{ddd} {hh:mm} to {ddd} {hh:mm}
{ddd} to {ddd} {hh:mm} to {hh:mm}
```

## *AKIPS API*

```
not {ddd} {hh:mm} to {hh:mm}

not {ddd} {hh:mm} to {ddd} {hh:mm}

not {ddd} to {ddd} {hh:mm} to {hh:mm}
  thishour

lasthour

today

yesterday

thisweek

lastweek

thismonth

lastmonth

thisyear

lastyear

last{duration}

from yyyy to yyyy

from yyyy-mm to yyyy-mm

from yyyy-mm-dd to yyyy-mm-dd

from yyyy-mm-dd hh:mm to yyyy-mm-dd hh:mm

from yyyy for {duration}

from yyyy-mm for {duration}

from yyyy-mm-dd for {duration}

from yyyy-mm-dd hh:mm for {duration}

from {timestamp} to {timestamp}

from {point in time} to {point in time}

from {point in time} to {point in time} [+ duration]

from {point in time} [+ duration] to

{point in time} [+ duration]
```

## *AKIPS API*

now

startoflastminute

endoflastminute

startofthisminute

endofthisminute

startoflasthour

endoflasthour

startofthishour

endofthishour

startofyesterday

endofyesterday

startoftoday

endoftoday

startoflastweek

endoflastweek

startofthisweek

endofthisweek

startoflastmonth

endoflastmonth

startofthismonth

endofthismonth

startoflastyear

endoflastyear

startofthisyear

endofthisyear

### 2.4.1 tf pairs and tf span

The `tf pairs` command will display a series of start, end pairs which indicate valid time ranges within the time filter.

The human readable form of this is shown in the `tf dump` command.

```
tf dump "lastweek; mon to fri 8:00 to 17:00"
tf pairs "lastweek; mon to fri 8:00 to 17:00"
```

The `tf span` command displays the total span of the time filter.

```
tf dump "lastweek; mon to fri 8:00 to 17:00"
tf span "lastweek; mon to fri 8:00 to 17:00"
```

### Examples

The past 30 minutes:

```
tf dump last30m

span 2021-03-06 09:20:50 to 2021-03-06 09:50:50
```

The past 24 hours:

```
tf dump last24h

span 2021-03-05 10:05:56 to 2021-03-06 10:05:56
```

The previous day (until midnight this morning):

```
tf dump last1d

span 2021-03-06 00:00:00 to 2021-03-06 23:59:59
```

The previous seven days (until midnight this morning):

```
tf dump last7d

span 2021-02-28 00:00:00 to 2021-03-06 23:59:59
```

During business hours for the past week:

```
tf dump "lastweek; mon to fri 8:00 to 17:00"

span 2021-02-24 00:00:00 to 2021-03-02 23:59:59
include 2021-02-25 08:00:00 to 2021-02-25 17:00:00
include 2021-02-26 08:00:00 to 2021-02-26 17:00:00
include 2021-02-27 08:00:00 to 2021-02-27 17:00:00
include 2021-02-28 08:00:00 to 2021-02-28 17:00:00
include 2021-03-01 08:00:00 to 2021-03-01 17:00:00
```

## AKIPS API

During trading hours (with Thursday-evening trading) for the past week:

```
tf dump "lastweek; mon to fri 8:00 to 17:00; thu 8:00 to 21:00"
```

```
span 2021-02-24 00:00:00 to 2021-03-02 23:59:59
include 2021-02-25 08:00:00 to 2021-02-25 17:00:00
include 2021-02-26 08:00:00 to 2021-02-26 17:00:00
include 2021-02-27 08:00:00 to 2021-02-27 17:00:00
include 2021-02-28 08:00:00 to 2021-02-28 21:00:00
include 2021-03-01 08:00:00 to 2021-03-01 17:00:00
```

Outside trading hours (with Saturday-morning trading) for the past week:

```
tf dump "lastweek; not mon to fri 8:00 to 17:00; not sat 8:00 to 12:00"
```

```
span 2021-02-24 00:00:00 to 2021-03-02 23:59:59
include 2021-02-24 00:00:00 to 2021-02-25 08:00:00
include 2021-02-25 17:00:00 to 2021-02-26 08:00:00
include 2021-02-26 17:00:00 to 2021-02-27 08:00:00
include 2021-02-27 17:00:00 to 2021-02-28 08:00:00
include 2021-02-28 17:00:00 to 2021-03-01 08:00:00
include 2021-03-01 17:00:00 to 2021-03-02 08:00:00
include 2021-03-02 12:00:00 to 2021-03-02 23:59:59
```

During business hours in the first quarter of 2021:

```
tf dump "2021 q1; mon to fri 8:00 to 17:00"
```

```
span 2021-01-01 00:00:00 to 2021-03-31 23:59:59
include 2021-01-01 08:00:00 to 2021-01-01 17:00:00
include 2021-01-02 08:00:00 to 2021-01-02 17:00:00
include 2021-01-03 08:00:00 to 2021-01-03 17:00:00
...
include 2021-03-28 08:00:00 to 2021-03-28 17:00:00
include 2021-03-29 08:00:00 to 2021-03-29 17:00:00
```

The past 30 minutes:

```
tf dump "from now - 1h to now - 30m"
```

```
span 2021-03-06 09:30:58 to 2021-03-06 10:00:58
```

Today until now:

```
tf dump "from startoftoday to now"
```

```
span 2021-03-06 00:00:00 to 2021-03-06 10:33:13
```

## AKIPS API

Business hours on 1 February 2021:

```
tf dump "2021-02-01 8:00 to 17:00"
span 2021-02-01 08:00:00 to 2021-02-01 17:00:00
```

After business hours on 1 February 2021:

```
tf dump "from 2021-02-01 18:00 to 2021-02-02 06:00"
span 2021-02-01 18:00:00 to 2021-02-02 06:00:00
```

The first two months of 2021:

```
tf dump "from 2021 for 2M"
span 2021-01-01 00:00:00 to 2021-03-01 00:00:00
```

Yesterday from midday for three hours:

```
tf dump "from startofyesterday + 12h to startofyesterday + 15h"
span 2021-03-05 12:00:00 to 2021-03-05 15:00:00
```

### 2.4.2 tf check

The `tf check` command will display either `1` (true) or `0` (false) depending on whether the given point in time (Unix epoch time) is within the time filter.

Use the following syntax:

```
tf check {point in time} "{time filter}"
```

### Examples

```
tf check 1594771234 "from 2020-07-05 to 2020-07-11"
0
ok: tf check 1594771234 "from 2020-07-05 to 2020-07-11"

tf check 1594771234 "from 2020-07-12 to 2020-07-18"
1
ok: tf check 1594771234 "from 2020-07-12 to 2020-07-18"
```

## 2.5 IP address filters

## AKIPS API

You can use IP address filters throughout AKIPS. Use the following syntax:

```
{address}/{mask}
```

```
{address}.*
```

```
{address}[{range}]
```

```
{address}[{range}]/{mask}
```

```
{address}[{range}].*
```

## Examples

A single IPv4 address:

```
10.1.1.50
```

```
10.1.1.50
```

All addresses in a single logical network (i.e. with a specified 24-bit prefix):

```
10.0.0.0/24
```

```
10.0.0.*
```

```
10.0.0.0  
10.0.0.1  
... 10.0.0.254  
10.0.0.255
```

A host ID in a range of 256 logical networks (within a specified range of 256 concurrent 24-bit blocks):

```
10.0.0-255.1
```

```
10.0.*.1
```

```
10.0.0.1  
10.0.1.1  
10.0.2.1  
... 10.0.255.1
```

A single IPv6 address:

```
fd00:10:1:1::1
```

```
fd00:10:1:1::1
```



## *AKIPS API*

Two hundred and fifty-six IPv6 addresses in a single logical network:

```
fd00:10:1:1::0-ff
```

```
fd00:10:1:1::0  
fd00:10:1:1::1  
... fd00:10:1:1::fe  
fd00:10:1:1::ff
```

A host ID in a range of 50 logical networks:

```
fd00:10:1:1-50:1
```

```
fd00:10:1:1::1  
fd00:10:1:2::1  
... fd00:10:1:50::1
```

## 3 Database overview

### 3.1 Entities

The ADB architecture defines three levels of entities:

- parents (see 3.1.1)
- childs (see 3.1.2)
- attributes (see 3.1.3).

You must create entities in the following order:

- create the **parent** before you add a **child**
- create the **child** before you add an **attribute**.

Each entity must have a type, which you cannot change once created.

Level	Entity	Type	Value	Notes
1	parent	Y	N	a parent is an independent item
2	child	Y	optional	a child is a component of its parent entity
3	attribute	Y	Y	an attribute is a component of its parent and child entities

### Example

To measure the flow of traffic (by counting the number of incoming bytes, or InOctets) for a particular router:

- the parent is a device
- the child is an interface
- the attribute is a counter.

## DATABASE OVERVIEW

You could define it as:

Level	Entity	Type	Value	Name
1	parent	device	router01	
2	child	interface	1, Test network	Se11/1
3	attribute	counter	1	IF-MIB.ifHC InOctets

### 3.1.1 Level 1: parent

Parent types:

- parent (generic)
- device
- user
- report.

### 3.1.2 Level 2: child

Child types:

- child (generic)
- interface
- ipsla
- memory
- processor
- storage
- system
- temperature.

A child may optionally have a value in the format of index,[description]

E.g. a child, Fa0/1, may be assigned a value of 1,Link to server:

- 1 is the ifIndex of the interface in ifTable
- Link to server is the ifAlias.

### 3.1.3 Level 3: attribute

Attribute types:

Type	Value	Example
counter	must be 1	1
enum	positive integer,text (enumerated list)	1, up 2, down
gauge	scale: integer (positive to multiply, negative to divide)	-2
integer	positive or negative whole number or 0	100287
RTT	positive integer (microseconds)	430
text	up to 2000 characters	the quick brown fox
timestamp	seconds since start of Unix epoch	1406787487
uptime	seconds since status change	13095

### 3.1.4 Virtual attributes

AKIPS doesn't poll virtual attributes but instead meaningfully interprets the data. E.g. ifutil is the in-rate divided by the interface speed.

Virtual attribute types:

Type	Description
ifutil	interface utilisation
ifrate	interface bps
vutil	utilisation
vnutil	calculate used from free
vdifff	difference $x = a - b$ temp_f convert C to F

## 3.2 Groups

When you create a group, specify the group type. You can assign an entity only to a group with the same type.

A child inherits group assignments from its parent.

An attribute inherits group assignments from both its child and parent.

### 3.2.1 Super groups

Use super groups (groups of groups) to create a hierarchy of your network.

You cannot assign an entity directly to a super group. You can assign only a group to a super group.

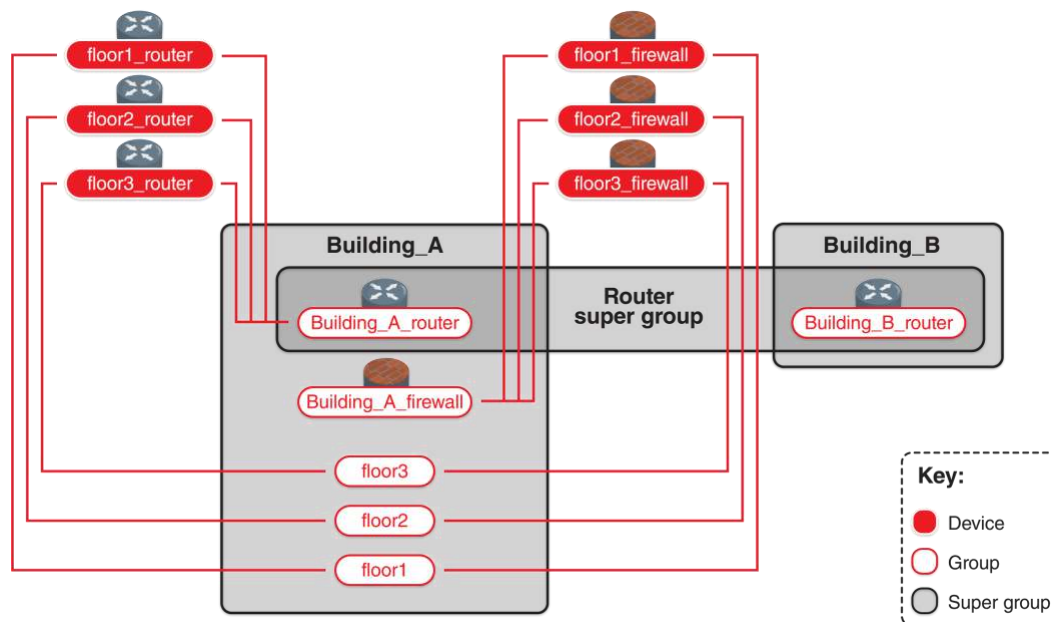
### Example

Assign a **router** on **floor 1** of **building A** to both:

- floor 1 group
- building A router group.

Further assign the **building A router** group to both:

- **building A** super group
- **router** super group.



Graphic 49a. Mapping a hierarchy of groups and super groups

### Edit a super group:

Go to **Admin > Grouping > Manual Grouping**.

Click **Super Groups**.

Select the relevant super group.

### Change the super group's name:

Overtyping the name.

Click **Rename**.

### Edit the super group's configuration:

Click **Edit**.

Select or deselect the relevant checkboxes

### Delete the super group:

Click **Delete**.

The screenshot displays the AKiPS web interface. The top navigation bar includes 'Dashboards', 'Reports', 'Tools', 'Admin', 'New', and 'PDF'. The user is logged in as 'admin'. The main content area is titled 'Manual Grouping' and contains a sidebar with a list of categories: CPUs, Devices, IPSLA, Interfaces, Memory, Netflow Exporters, Storage, Super Groups, and Temperature. The 'Super Groups' category is selected. The main panel shows a list of super groups: '0-Airport' (selected), '7-EMEA', and 'Region-A'. The '0-Airport' group is expanded, showing a list of sub-groups with checkboxes: '0-AU-VIC-BAL', '0-AU-VIC-GEE', '0-AU-VIC-MEL', '0-Building-3', '0-Building-4', '0-Melbourne', '0-MyGov', '0-Sydney' (checked), '0-Test', '1-Building-3', '1-Building-4', '1-Building-16', '1-Fraser', '1-Network-A', '1-Network-B', and '1-Notre-Dame-Test'. Buttons for 'Save', 'Help', 'Grouping Rules', and 'Delete Broken Rules' are visible.

Graphic 49. Editing a super group's configuration

### 3.2.2 Configuring groups

Use the web API to manually:

- add or delete a group or super group
- assign an entity to a group
- assign a group to a super group
- remove an entity from a group
- remove a group from a super group.

#### Examples

##### CPU

Add a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=processor;group={group_name}; mode=add"
```

Assign an entity to a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=processor;group={group_name};  
mode=assign;device={device_name};child={child_name}"
```

Remove an entity from a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=processor;group={group_name};  
mode=clear;device={device_name};child={child_name}"
```

Delete a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=processor;group={group_name}; mode=delete"
```

##### Device

Add a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=device;group={group_name}; mode=add"
```

Assign an entity to a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=device;group={group_name};  
mode=assign;device={device_name}"
```

## *DATABASE OVERVIEW*

Remove an entity from a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=device;group={group_name};  
mode=clear;device={device_name}"
```

Delete a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=device;group={group_name}; mode=delete"
```

## **IPSLA**

Add a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=ipsla;group={group_name};  
mode=add"
```

Assign an entity to a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=ipsla;group={group_name};  
mode=assign;device={device_name};child={child_name}"
```

Remove an entity from a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=ipsla;group={group_name};  
mode=clear;device={device_name};child={child_name}"
```

Delete a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=ipsla;group={group_name}; mode=delete"
```

## **Interface**

Add a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=interface;group={group_name}; mode=add"
```

Assign an entity to a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=interface;group={group_name};  
mode=assign;device={device_name};child={child_name}"
```



## DATABASE OVERVIEW

Remove an entity from a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=interface;group={group_name};  
mode=clear;device={device_name};child={child_name}"
```

Delete a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=interface;group={group_name}; mode=delete"
```

### Memory

Add a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=memory;group={group_name}; mode=add"
```

Assign an entity to a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=memory;group={group_name};  
mode=assign;device={device_name};child={child_name}"
```

Remove an entity from a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=memory;group={group_name};  
mode=clear;device={device_name};child={child_name}"
```

Delete a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=memory;group={group_name}; mode=delete"
```

### NetFlow

Add a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=flow;group={group_name};  
mode=add"
```

Assign an entity to a group:

```
curl -s Error! Hyperlink reference not valid.
```

Remove an entity from a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=flow;group={group_name};  
mode=clear;device={device_name};child={child_name}"
```

## *DATABASE OVERVIEW*

Delete a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=flow;group={group_name}; mode=delete"
```

### **Storage**

Add a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=storage;group={group_name}; mode=add"
```

Assign an entity to a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=storage;group={group_name};  
mode=assign;device={device_name};child={child_name}"
```

Remove an entity from a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=storage;group={group_name};  
mode=clear;device={device_name};child={child_name}"
```

Delete a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=storage;group={group_name}; mode=delete"
```

### **Super group**

Add a super group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=super; group={super_group_name};mode=add"
```

Assign a group to a super group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=group;group={group_name};  
mode=assign;device={group_name}"
```

Remove a group from a super group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=group;group={group_name};  
mode=clear;device={group_name}"
```

## DATABASE OVERVIEW

Delete a super group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=super;  
group={super_group_name};mode=delete"
```

### Temperature

Add a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=temperature;  
group={group_name};mode=add"
```

Assign an entity to a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=temperature;  
group={group_name};mode=assign;device={device_name}; child={child_name}"
```

Remove an entity from a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=temperature;  
group={group_name};mode=clear;device={device_name}; child={child_name}"
```

Delete a group:

```
curl -s "http://{server}/api-script?password={pwd};  
function=web_manual_grouping;type=temperature;  
group={group_name};mode=delete"
```

## 3.3 Events records

AKIPS stores events records in chronologically ordered blocks.

After 72 hours, AKIPS compresses and archives events records. You will not be able to edit or delete these.

Events records may include:

- enumeration changes
- uptime resets
- threshold events.

Each record in the events database includes:

- event time (epoch timestamp)
- entity (parent, child or attribute)
- flags (critical, enum, threshold and uptime)

## DATABASE OVERVIEW

- value.

### To view enumeration and uptime attributes:

**Go to Admin > Alerting > Status Alerts.**

Scroll down the right-hand side of the page to find the table of `Status Attributes`.

### To view threshold attributes:

**Go to Admin > Alerting > Threshold Alerts.**

Scroll down the right-hand side of the page to find the table of `Threshold Attributes`.

## 3.4 Time series

The features for time series include:

- three years of historical data
- 60-second values (no rollups or averages of stored data)
- 30-day blocks.

Rolling averages are available for the past five minutes to 30 days.

Time-series data types are:

- counters
- gauges
- RTT (microseconds).

## 4 Config and events

### 4.1 Entity commands

#### 4.1.1 add

Use the `add` command to create a new entity. If the entity already exists, AKIPS will update it. You cannot change an entity's type once created.

Use the following syntax:

```
add {type} {parent name}
add {type} {parent name} {child name} = [{index},[description]]
add {type} {parent name} {child name} {attribute name}
[= {value}]
```

### Examples

Add a router with an interface:

```
add device Router1

add system Router1 sys = 0

add text Router1 sys SNMPv2-MIB.sysName = "Router1.akips.com"
add text Router1 sys SNMPv2-MIB.sysLocation = "Test Rack"
add text Router1 sys SNMPv2-MIB.sysDesc = "Cisco IOS 7200
Version 12.4(9) T RELEASE (fc1)"

add text Router1 sys SNMPv2-MIB.sysContact = "AKIPS developers"
add uptime Router1 sys SNMPv2-MIB.sysUpTime = 1432703430
add text Router1 sys ip4addr = 10.1.8.250

add text Router1 sys ip6addr = fd00:10:1:8::250
  add system Router1 ping4 = "4,10.1.8.4"
add rtt Router1 ping4 PING.icmpRtt
add counter Router1 ping4 PING.icmpDup
add counter Router1 ping4 PING.icmpLost

add enum Router1 ping4 PING.icmpState = 2,up
add system Router1 ping6 = "6,fd00:10:1:8::250"
  add rtt Router1 ping6 PING.icmpRtt
add counter Router1 ping6 PING.icmpDup
```

## CONFIG AND EVENTS

```
add counter Router1 ping6 PING.icmpLost
add enum Router1 ping6 PING.icmpState
add interface Router1 Fa0/0 = "1,Link to remote site"
  add integer Router1 Fa0/0 IF-MIB.ifSpeed = 1000000000

add text Router1 Fa0/0 IF-MIB.ifAlias = "Link to remote site"

add counter Router1 Fa0/0 IF-MIB.ifHCInOctets = 1
add counter Router1 Fa0/0 IF-MIB.ifHCOutOctets = 1
add enum Router1 Fa0/0 IF-MIB.ifAdminStatus = 1,up
add enum Router1 Fa0/0 IF-MIB.ifOperStatus = 1,up
add counter Router1 Fa0/0 IF-MIB.ifHCInBroadcastPkts = 1
add counter Router1 Fa0/0 IF-MIB.ifHCOutBroadcast Pkts = 1
add counter Router1 Fa0/0 IF-MIB.ifHCInMulticastPkts = 1
add counter Router1 Fa0/0 IF-MIB.ifHCOutMulticast Pkts = 1
```

Add some custom entries for asset information:

```
add child Router1 asset
add text Router1 asset serial_number = 123456789
add text Router1 asset purchase_date = 2017/05/02
add integer Router1 asset purchase_cost = 12013
```

### 4.1.2 delete

Use the `delete` command to remove a single entity, including all of its childs and attributes.

Use the following syntax:

```
delete {type} {parent name} [{child name}] {attribute name}]
```

### Examples

Delete a device:

```
delete device CoreRouter12
```

Delete a child:

```
delete interface CoreRouter1 Fa0/03
```

Delete an attribute:

```
delete integer CoreRouter1 Fa0/0 IF-MIB.ifSpeed
```

### 4.1.3 get

Use the `get` command to return the value assigned to a child or attribute.  
Use the following syntax:

```
get {parent name} {child name} [{attribute name}]
```

#### Examples

Get the value for an interface:

```
get swt250 Fa0/1  
10001,Link to swt2512
```

Get the value for an attribute:

```
get swt250 sys SNMPv2-MIB.sysLocation  
Test room
```

The `get` command for an attribute with a type of uptime returns two values separated by a comma:

1. when the value was reset to zero (epoch timestamp)
2. when the attribute value was last updated (epoch timestamp).

#### Example

```
get swt250 sys SNMPv2-MIB.sysUpTime  
1427283250,1433815768
```

The `get` command for an attribute with a type of enum returns five values separated by commas:

1. list number (from MIB)
2. text value (from MIB)
3. time created (epoch timestamp)
4. time modified (epoch timestamp)
5. child description.

#### Example

```
get swt250 Fa0/1 IF-MIB.ifOperStatus  
1,up,1431579447,1431579447,link to swt251
```

#### 4.1.4 mdelete

Use the `mdelete` command to remove several entities at once. You can test your selection first with `mget` (see 4.1.5).

You can use `regex` to precisely target the entities.  
Use the following syntax:

```
mdelete {type} {parent regex} [{child regex}  
[{attribute regex}]]
```

#### Examples

Delete all devices (We do not recommend this!):

```
mdelete device *
```

Delete all devices in a group:

```
mdelete device * any group testlab
```

Delete all IF-MIB attributes:

```
mdelete * CoreRouter1 Fa0/0 /IF-MIB/
```

Delete all ipsla entities for a device:

```
mdelete * Corerouter1 /ipsla/
```

#### 4.1.5 mget

Use the `mget` command to retrieve large amounts of configuration information.

Use the following syntax:

```
mget {type}
```

#### Examples

```
mget device mget user mget interface mget processor  
mget system
```

You can also add `regex`.

Use the following syntax:

```
[{parent regex} [{child regex} [{attribute regex}]]]  
[value {text|/regex/|integer|ipaddr}] [profile {profile name}] [any|all|not  
group {group name}] ...
```

The format of the return values depends on which entity (parent, child or attribute) you request.



## CONFIG AND EVENTS

If you request a parent then the type must be a parent or a wildcard, e.g.

```
mget * /^nyc-.*-rtr/ mget device *
mget user *
```

If you request a child then the type must be a child or a wildcard, e.g.

```
mget * swt250 /^Fa/ mget system swt250 * mget interface * *
mget interface * /^Fa0/1/
```

If you request an attribute then the type must be an attribute or a wildcard, e.g.

```
mget * swt250 * *

mget counter swt250 Fa0/1 /IF-MIB/
```

## Examples

Retrieve the entire configuration for a single device:

```
mget * swt250 * *
```

Retrieve the sysName for each device:

```
mget text * * SNMPv2-MIB.sysName
```

```
Atlanta-ro sys SNMPv2-MIB.sysName = Atlanta-ro.akips.com Baltimore-ro sys
SNMPv2-MIB.sysName = Baltimore-ro.akips.com Boston-ro sys SNMPv2-MIB.sysName
= Boston-ro.akips.com
Charlotte-ro sys SNMPv2-MIB.sysName = Charlotte-ro.akips.com cisco-74-1-1 sys
SNMPv2-MIB.sysName = cisco-74-1-1
```

Retrieve all IF-MIB objects for a single interface:

```
mget * swt250 Fa0/1 /^IF-MIB.*/

swt250 Fa0/1 IF-MIB.ifAdminStatus = 1,up,1434416774,1434416774
swt250 Fa0/1 IF-MIB.ifAlias = Link to swt251
swt250 Fa0/1 IF-MIB.ifDescr = FastEthernet0/1
swt250 Fa0/1 IF-MIB.ifDuplex = 3,fullDuplex,1434416774,1434416774
swt250 Fa0/1 IF-MIB.ifHCInBroadcastPkts = 1
```

Retrieve the IPv4/6 addresses for each device:

```
mget text * sys /ip.addr/

Atlanta-ro sys ip4addr = 10.4.26.1
Atlanta-ro sys ip6addr = fd00:10:4:26::1 Baltimore-ro sys ip4addr = 10.4.22.1
Baltimore-ro sys ip6addr = fd00:10:4:22::1 Boston-ro sys ip4addr = 10.4.23.1
Boston-ro sys ip6addr = fd00:10:4:23::1
...
```

### Case study

A customer used the `mget` command to obtain the serial numbers of all devices on his network, including their access points:

```
mget * * * /serial/
```

#### 4.1.6 mlist

Use the `mlist` command to retrieve a list of matching entities.

Unlike the `mget` command, the output does not contain any entity values. Use the following syntax:

```
mlist {type} [{parent regex} [{child regex} [{attribute regex}]]]  
[value {text|regex|integer|ipaddr}] [profile {profile name}] [any|all|not  
group {group name}] ...
```

### Examples

Retrieve a list of all interfaces on a device:

```
mlist interface swt250 * swt250 Fa0/1  
swt250 Fa0/10 swt250 Fa0/11 swt250 Fa0/12 swt250 Fa0/16  
...
```

Retrieve a list of interface attributes on a device with the type counter:

```
mlist counter swt250 * /^IF-MIB./  
swt250 Fa0/1 IF-MIB.ifHCInBroadcastPkts swt250 Fa0/1 IF-  
MIB.ifHCInMulticastPkts swt250 Fa0/1 IF-MIB.ifHCInOctets swt250 Fa0/1 IF-  
MIB.ifHCInUcastPkts  
swt250 Fa0/1 IF-MIB.ifHCOutBroadcastPkts swt250 Fa0/1 IF-  
MIB.ifHCOutMulticastPkts swt250 Fa0/1 IF-MIB.ifHCOutOctets swt250 Fa0/1 IF-  
MIB.ifHCOutUcastPkts swt250 Fa0/1 IF-MIB.ifInDiscards  
swt250 Fa0/1 IF-MIB.ifInErrors  
...
```

Retrieve a list of enumerated attribute types configured for a device:

```
mlist enum swt250 * *  
  
swt250 cstack.1 CISCO-STACK-MIB.moduleStandbyStatus swt250 cstack.1 CISCO-  
STACK-MIB.moduleStatus swt250 Fa0/1 IF-MIB.ifAdminStatus  
swt250 Fa0/1 IF-MIB.ifDuplex swt250 Fa0/1 IF-MIB.ifOperStatus swt250 Fa0/1  
IF-MIB.ifType  
swt250 Fa0/10 IF-MIB.ifAdminStatus  
...
```

#### 4.1.7 mtype

Use the `mtype` command to retrieve a list of matching entities.

Unlike the `mget` command, the output contains entity types, not values. Use the following syntax:

```
mtype {type} [{parent regex} [{child regex} [{attribute regex}]] [value  
{text|regex|integer|ipaddr}] [profile {profile name}] [any|all|not group  
{group name}] ...]
```

## CONFIG AND EVENTS

### Examples

Retrieve a list of childs for a device:

```
mtype * swt250 *

swt250 cpu.1 processor swt250 cstack.1 child swt250 Fa0/1 interface
...
swt250 Gi0/1 interface swt250 Gi0/2 interface swt250 ipsla.1 ipsla swt250
ipsla.2 ipsla swt250 ping6 system swt250 psu.1003 system swt250 ram.1 memory
swt250 ram.2 memory swt250 sys system
```

Retrieve a list of attributes for a device:

```
mtype * swt250 * *

swt250 cpu.1 CISCO-PROCESS-MIB.cpmCPUTotal1minRev gauge swt250 cstack.1
CISCO-STACK-MIB.moduleModel text
swt250 cstack.1 CISCO-STACK-MIB.moduleStandbyStatus enum swt250 cstack.1
CISCO-STACK-MIB.moduleStatus enum swt250 Fa0/1 IF-MIB.ifAdminStatus enum
swt250 Fa0/1 IF-MIB.ifAlias text swt250 Fa0/1 IF-MIB.ifDescr text swt250 Fa0/1
IF-MIB.ifDuplex enum
swt250 Fa0/1 IF-MIB.ifHCInBroadcastPkts counter swt250 Fa0/1 IF-
MIB.ifHCInMulticastPkts counter
...
```

#### 4.1.8 rename

Use the `rename` command to rename a parent or child.

Use the following syntax:

```
rename {type} {existing parent name} {new parent name}

rename {type} {existing parent name} {existing child name}
{new parent name} {new child name}
```

### Examples

Rename a device:

```
rename device nyc-rtr usa-nyc-rtr12
```

Move an interface from one device to another:

```
rename interface nyc-rtr1 Se0/1 nyc-rtr2 Se0/1
```

#### 4.1.9 set

Use the `set` command to update an entity's value.

Use the following syntax:

```
set {parent name} {child name} [{attribute name}] = {value}
```

## Examples

```
set swt250 sys SNMPv2-MIB.sysLocation = "AKIPS Test Lab" set swt250 sys
SNMPv2-MIB.sysContact = "AKIPS developers" set swt250 sys ip4addr =
10.1.8.250
```

## 4.2 Group commands

### 4.2.1 add

Use the `add` command to create a new group.

If the group already exists, AKIPS will not update it.  
You cannot change a group's type once created.  
Use the following syntax:

```
add {type} group {group name}
```

## Examples

Add a group:

```
add device group core-routers
```

Add an interface group:

```
add interface group uplinks
```

Add an ipsla group:

```
add ipsla group Jitter-tests
```

Add device and super groups:

```
add device group SFO-BuildingA-Fl01 add device group SFO-BuildingA-Fl02 add
device group SFO-BuildingA-Fl03
...
```

```
add device group SFO-BuildingB-Fl01 add device group SFO-BuildingB-Fl02 add
device group SFO-BuildingB-Fl03
...
```

```
add super group SFO-BuildingA add super group SFO-BuildingB add super group
SFO-Campus
```

Add interface and super groups:

```
add interface group SFO-BuildingA-switch-uplinks
```

```
...
```

```
add super group SFO-Campus-switch-uplinks
```

## CONFIG AND EVENTS

You can also use the `add` command to create a new profile group. Use the following syntax:

```
add profile group {group name}
```

### Example

```
add profile group WAN-Tech-Support
```

### 4.2.2 assign

Use the `assign` command to assign groups.

Use the following syntax:

```
assign group {group name} = {super group name}
```

```
assign {type} {parent regex} [{child regex} [{attribute regex}]]  
[value {text|integer|/regex|/ipaddr}] [profile {profile name}] [any|all|not  
group {group name} ...] = {target group} ...
```

### Examples

Create vendor groups and assign devices to them based on attribute values:

```
add device group Cisco add device group Extreme  
assign * * sys SNMPv2-MIB.sysDescr value /Cisco/ = Cisco  
assign * * sys SNMPv2-MIB.sysDescr value /Cabletron/ = Extreme  
assign * * sys SNMPv2-MIB.sysDescr value /Enterasys/ = Extreme  
assign * * sys SNMPv2-MIB.sysDescr value /Extreme/ = Extreme
```

Put different models into groups using `sysDescr`:

```
add device group Cisco-3600  
add device group Cisco-3700  
add device group Cisco-7200  
assign * * sys SNMPv2-MIB.sysDescr value "/Cisco IOS 3600/" =  
Cisco-3600  
  
assign * * sys SNMPv2-MIB.sysDescr value "/Cisco IOS 3700/" =  
Cisco-3700  
  
assign * * sys SNMPv2-MIB.sysDescr value "/Cisco IOS 7200/" = Cisco-7200
```

Put different models into groups using `sysObjectID`:

```
add device group Cisco-3600  
add device group Cisco-3700  
add device group Cisco-7200  
assign * * sys SNMPv2-MIB.sysObjectID value  
/CISCO-PRODUCTS-MIB.cisco36/ = Cisco-3600  
  
assign * * sys SNMPv2-MIB.sysObjectID value  
/CISCO-PRODUCTS-MIB.catalyst37xxStack/ = Cisco-3700
```

## CONFIG AND EVENTS

```
assign * * sys SNMPv2-MIB.sysObjectID value  
/CISCO-PRODUCTS-MIB.cisco72/ = Cisco-7200
```

Create some device groups:

```
add device group routers  
add device group switches  
add device group core-routers  
add device group regional-routers  
add device group core-switches  
add device group regional-switches
```

Create some interface groups:

```
add interface group core-links  
  
add interface group core-switch-uplinks  
add interface group regional-links  
add interface group servers
```

Assign interfaces to a group:

```
add interface group serial-links  
  
assign interface * /^Se/ = serial-links27
```

Assign all links with a specific ifAlias to a group:

```
add interface group Switch-Uplinks  
  
assign interface * * IF-MIB.ifAlias value /uplink/ = Switch-Uplinks
```

Create a group hierarchy:

```
add device group central-office-fl01  
  
assign device /central-office-fl01-.*/ = central-office-fl01  
add device group central-office-fl02  
assign device /central-office-fl02-.*/ = central-office-fl02  
add super group central-office  
assign group central-office-fl01 = central-office assign group central-office-  
fl02 = central-office  
add super group Brisbane  
assign group central-office = Brisbane  
add super group Queensland  
assign group Brisbane = Queensland  
add super group Australia  
assign group Queensland = Australia
```

## CONFIG AND EVENTS

You can also use the `assign` command to assign a user or group to a profile group. Use the following syntax:

```
assign user {user name} = {profile name}

assign {type} group {group regex} = {profile name}
assign group {group regex} = {profile name}
```

### Examples

Allow a profile to access a device group:

```
assign device group Alcatel = Support
```

Allow a profile to access an interface group:

```
assign interface group ifspeed_10G = Support
```

Assign a user to a location profile:

```
assign user a.jager = Building02
```

Create a profile group and allow it to access specific devices:

```
add profile group WAN-Tech-Support
add device group WAN-Devices
assign * * sys SNMPv2-MIB.sysDescr value /Cisco/ = WAN-Devices
add report group WAN-Reports
assign * * config vendor value /Cisco/ = WAN-Reports
    assign group WAN-Devices = WAN-Tech-Support
assign group WAN-Reports = WAN-Tech-Support
```

Create a profile group, assign a user and allow access to devices with a specific sysName:

```
add profile group Helpdesk-Profile
assign user fred = Helpdesk-Profile
add device group Helpdesk-Devices
assign * /-swt|-rtr/ = Helpdesk-Devices
assign group helpdesk-devices = Helpdesk-Profile
```

### 4.2.3 clear

Use the `clear` command to remove group assignments.

Use the following syntax:

```
clear group {group name} = {super group name} clear {type}
{parent regex} [{child regex} [{attribute regex}]]
[value {text|/regex/|integer|ipaddr}] [profile {profile name}] [any|all|not
group {group name} ...] = {target group} ...
```

### Example

Assign devices to a group and then clear switches:

```
assign * * sys SNMPv2-MIB.sysDescr value /Cisco/ = Cisco
clear * * sys SNMPv2-MIB.sysDescr value "/Cisco IOS C3560/"
= Cisco
```

You can also use the `clear` command to remove entities from a profile group.

Use the following syntax:

```
clear user {user name} = {profile name}
clear group {group name} = {profile name} clear group {group regex} =
{profile name}
clear {type} group {group regex} = {profile name}
```

### Examples

Remove a user from a profile group:

```
clear user a.jager = Cisco-Only
```

Remove a profile group's access to a device group:

```
clear group WAN-Device = WAN-Tech-Support
```

#### 4.2.4 count

Use the `count` command to return the number of entities in a group.

Use the following syntax:

```
count {type} group {group regex}
```

### Example

Count the number of interfaces in a group:

```
count interface group uplinks
```

#### 4.2.5 delete

Use the `delete` command to remove a single group. Use the following syntax:

```
delete {type} group {group name}
```

### Examples

Delete a device group:

```
delete device group core-routers
```

Delete an interface group:

```
delete interface group uplinks
```



## CONFIG AND EVENTS

Delete a super group:

```
delete super group SFO-Campus
```

### 4.2.6 get

Use the `get` command to check if a group exists. Use the following syntax:

```
get group {group name}
```

#### Example

```
get group Cisco device
```

You can also use the `get` command to check if a profile group exists.

Use the following syntax:

```
get profile user {user name}
```

#### Example

Show the profile allocated to a user:

```
get profile user a.jager
```

[Support](#)

### 4.2.7 list

Use the `list` command to see the groups in a specific type.

Use the following syntax:

```
list {type} group [{parent name} [{child name} [{attribute name}]]] [profile  
{profile name}]  
[super {super group name}]
```

#### Examples

```
list profile group
```

```
list device group
```

```
list * group
```

You can also use the `list` command to find the profile groups associated with an entity.

Use the following syntax:

```
list profile {type} {parent name} [{child name}]  
{attribute name}]
```

#### Examples

Find the groups associated with a profile group:

## CONFIG AND EVENTS

```
list * group profile Support
```

```
super,Australia super,Brisbane super,central-office device,Cisco device,Dell
interface,ifspeed_1 interface,ifspeed_1.4G interface,ifspeed_1.5M
interface,ifspeed_1.8
```

Find the devices associated with a profile group:

```
list device group profile Support
```

```
Cisco
Dell
```

Find the interfaces associated with a profile group:

```
list interface group profile Support
ifspeed_1
ifspeed_1.4G
ifspeed_1.8G ifspeed_100G
...
```

Find all of the super groups that a profile group can access:

```
list device super group profile Support
Australia
Brisbane
central-office
```

### 4.2.8 mget

Use the `mget` command to retrieve large amounts of configuration information for a profile group. Use the following syntax:

```
mget {type} [{parent regex} [{child regex} [{attribute regex}]]]
[value {text|integer|regex|ipaddr}] [profile {profile name}] [any|all|not
group {group name} ...]
```

### Example

Show every device that a profile group can access:

```
mget device profile Cisco-Only
cisco-132-0-1
cisco-132-0-10
```

### 4.2.9 mlist

Use the `mlist` command to find entities associated with a profile group. Use the following syntax:

```
mlist {type} [profile {profile name}]
```

## Example

Show which devices a profile group can access:

```
mlist device profile fred
```

### 4.2.10 mtype

Use the `mtype` command to return an entity type for a profile group.

Use the following syntax:

```
mtype {type} [profile {profile name}]
```

## Example

Show which entity types a profile group can access:

```
mtype * profile Fred
```

### 4.2.11 prune

Use the `prune` command to remove empty groups.

Use the following syntax:

```
prune {type} group [{group regex}]
```

## Examples

Remove all empty device groups:

```
prune device group
```

Remove all empty interface groups:

```
prune interface group
```

### 4.2.12 rename

Use the `rename` command to rename a group. This will retain all of its associations.

Use the following syntax:

```
rename {type} group {existing group name} {new group name}
```

## Examples

Rename a profile group:

```
rename profile group Support Network-Support
```

Rename a device group:

```
rename device group core-routers USA-Core-Routers
```

## CONFIG AND EVENTS

Rename an interface group:

```
rename interface group uplinks Switch-Uplinks
```

### 4.3 Event commands

Event flags include:

- **ack**
- **critical**. AKIPS automatically adds a critical event flag to entities which match `crit_event`
- **suppress**
- **warning**. AKIPS automatically adds a warning event flag to entities which match `warn_event`
- **above**
- **below**.

#### 4.3.1 add

Use the `add` command to add an entity before creating an event.

You can use only the critical event flag with `add`.

Use the following syntax:

```
add event {time} {parent name} {child name} {attribute name}  
[{{event flags}}] = {value}
```

### Examples

Add `ifOperStatus` down and up events:

```
add event 0 swt250 Se0/1 IF-MIB.ifOperStatus = down  
add event 0 swt250 Se0/1 IF-MIB.ifOperStatus = up
```

Add ping-down and SNMP-down events:

```
add event 0 swt250 ping4 PING.icmpState = down  
add event 0 swt250 ping6 PING.icmpState = down  
add event 0 swt250 sys snmp.snmpState = down
```

Create a site-specific admin state for a device which may be set by another system:

```
add enum swt250 sys admin_state  
  
add event 0 swt250 sys admin_state = maintenance  
add event 0 swt250 sys admin_state = online
```

#### 4.3.2 clear

Use the `clear` command to clear a flag from an event.

Use the following syntax:

## CONFIG AND EVENTS

```
clear event {time} {parent name} {child name} {attribute name} =  
{event flags}
```

### Example

Clear the critical flag on a ping-down event:

```
clear event 1435822220 swt250 ping4 ping.icmpState = critical
```

### 4.3.3 delete

Use the `delete` command to delete an event.

Use the following syntax:

```
delete event {time} {parent name} {child name} {attribute name}
```

### Example

Delete a down event:

```
delete event 1435822225 swt250 ping4 PING.icmpState
```

### 4.3.4 mdelete

Use the `mdelete` command to delete all events for an attribute within a specific time range.

You can test your selection first with `mget` (see 4.3.5).

Use the following syntax:

```
mdelete event time {time filter} [{parent regex}  
[{child regex} [{attribute regex}]]] [profile {profile name}] [any|all|not  
group {group name} ...]
```

### Example

Delete all IF-MIB.ifOperStatus events for devices in a group for yesterday:

```
mdelete event time yesterday * * IF-MIB.ifOperStatus any group  
Edge-Switches
```

### 4.3.5 mget

Use the `mget` command to retrieve events records in chronological order. The child description will display only if you include one in its configuration.

Use the following syntax:

```
mget event {all,critical,enum,threshold,uptime}  
time {time filter} [{parent regex} {child regex}  
{attribute regex}] [profile {profile name}]  
[any|all|not group {group name} ...]
```

### Enumerated events

Use the following syntax:

## CONFIG AND EVENTS

```
{epoch} {parent} {child} {attribute} enum {flags} {value}  
[{child description}]
```

### Uptime events

Use the following syntax:

```
{epoch} {parent} {child} {attribute} uptime {flags}  
{last uptime in seconds} [{child description}]
```

### Threshold events

Use the following syntax:

```
{epoch} {parent} {child} {attribute} threshold {flags}  
{rule exceeded} [{child description}]
```

## Examples

Retrieve all ping-outage events for a router for the past hour:

```
mget event enum time last1h Columbus-ro /ping/ *  
  
1435894798 Columbus-ro ping4 PING.icmpState enum none down  
10.4.1.22  
1435894799 Columbus-ro ping6 PING.icmpState enum none down fd00:10:4:1::22  
1435895128 Columbus-ro ping4 PING.icmpState enum none up  
  
10.4.1.22  
1435895129 Columbus-ro ping6 PING.icmpState enum none up fd00:10:4:1::22
```

Retrieve all ifOperStatus events for the past day:

```
mget event enum time last1d * * IF-MIB.ifOperStatus  
  
1435846269 NewYork-ro Se2/2 IF-MIB.ifOperStatus enum none down  
Link to San Francisco  
1435846509 NewYork-ro Se2/2 IF-MIB.ifOperStatus enum none up  
Link to San Francisco  
1435848305 Chicago-ro Se1/4 IF-MIB.ifOperStatus enum none down  
Link to Dallas  
1435848365 Chicago-ro Se1/4 IF-MIB.ifOperStatus enum none up  
Link to Dallas
```

Retrieve all sysUpTime resets:

```
mget event uptime time last1d  
  
1435846486 Toronto-ro sys SNMPv2-MIB.sysUpTime uptime none 19044 1435848309  
Columbus-ro sys SNMPv2-MIB.sysUpTime uptime none 7863 1435849250 Detroit-ro  
sys SNMPv2-MIB.sysUpTime uptime none 10363 1435849830 Cleveland-ro sys  
SNMPv2-MIB.sysUpTime uptime none 25704
```

## CONFIG AND EVENTS

Retrieve all threshold events for the past hour:

```
mget event threshold time last1h
```

```
1436104800 cisco-74-1-19 cpu.2 CISCO-PROCESS-MIB.cpm CPUTotal
1minRev threshold critical,above last5m,avg,60
1436104800 Chicago-ro ping4 PING.icmpRtt threshold critical,below
last30m,avg,40000
1436104800 cisco-74-1-38 cpu.26 CISCO-PROCESS-MIB.cpm CPUTotal
1minRev threshold critical,above last5m,avg,60 1436105101 SanFrancisco-ro
ping4 PING.icmpRtt threshold critical,above last30m,avg,40000
1436105101 cisco-74-1-17 cpu.4 CISCO-PROCESS-MIB.cpm CPUTotal
1minRev threshold critical,below last5m,avg,60
1436105101 cisco-74-1-29 cpu.2 CISCO-PROCESS-MIB.cpm CPUTotal1min
threshold critical,above last5m,avg,60
1436105101 cisco-74-1-30 cpu.2 CISCO-PROCESS-MIB.cpm CPUTotal1min
threshold critical,above last5m,avg,60
1436105101 NewYork-ro ping6 PING.icmpRtt threshold critical,above
last30m,avg,40000
1436105101 NewYork-ro ping4 PING.icmpRtt threshold critical,above
last30m,avg,40000
1436105101 Chicago-ro ping4 PING.icmpRtt threshold critical,above
last30m,avg,40000
```

### 4.3.6 set

Use the `set` command to set event flags. Use the following syntax:

```
set event {time} {parent name} {child name} {attribute name} =
{event flags}
```

#### Example

Set a critical flag on an existing ping-down event:

```
set event 1435822220 swt250 ping4 ping.icmpState = critical
```

### 4.3.7 tget

Use the `tget` command to return time-series values for a number of events per interval.

Use the following syntax:

```
tget event {all,critical,enum,threshold,uptime} {interval secs}
time {time filter} [{parent regex} [{child regex}
[{attribute regex}]]] [profile {profile name}]
[any|all|not group {group name} ...]
```

#### Example

Retrieve time-series values for all ping events for yesterday in 24 one-hour intervals:

```
tget event all 3600 time yesterday * /ping/  
506,426,460,458,440,760,315,301,232,421,332,288,196,299,380,381,  
495,448,497,570,386,430,362,530
```

## 4.4 Time-series commands

### 4.4.1 calc

Use the `calc` command to calculate a total, average or median value.

Use the following syntax:

```
calc total|avg|{median NN} time {time filter}  
{type} {parent name} {child name} {attribute regex}  
[profile {profile name}] [any|all|not group {group name} ...]
```

### Examples

Calculate the total InOctets for a switch interface for yesterday:

```
calc total time yesterday counter swt250 Fa0/1 /InOctets/
```

Calculate the average InOctets for a switch interface for yesterday:

```
calc avg time yesterday counter swt250 Fa0/1 /InOctets/
```

### 4.4.2 mcalc

Use the `mcalc` command to calculate multiple total, average or median values.

Use the following syntax:

```
mcalc total|avg|{median NN} time {time filter}  
{type} {parent regex} {child regex} {attribute regex}  
[profile {profile name}] [any|all|not group {group name} ...]
```

### Example

Calculate the In/OutOctets for all interfaces in a group:

```
mcalc total time yesterday counter * * /^IF-MIB.*InOctets|  
^IF-MIB.*OutOctets/ any group wan-links mcalc avg time last1m i  
futil swt5 Gi2/0/1 *
```

```
Chicago-ro Se1/0 IF-MIB.ifInOctets = 90922240 Chicago-ro Se1/0 IF-  
MIB.ifOutOctets = 112385280 Chicago-ro Se1/1 IF-MIB.ifInOctets = 0  
Chicago-ro Se1/1 IF-MIB.ifOutOctets = 0  
Chicago-ro Se1/2 IF-MIB.ifInOctets = 11374963 Chicago-ro Se1/2 IF-  
MIB.ifOutOctets = 10454049 Chicago-ro Se1/3 IF-MIB.ifInOctets = 11409152
```

### Case study

A customer used the `mcalc` command to export the utilisation data (bits in or out) from groups of interfaces for specific device groups, for the previous five minutes as an average:



## CONFIG AND EVENTS

```
mcalc avg time last5m ifrate * * /IF-MIB.if.*BitRate/  
any group Circuit
```

### 4.4.3 mget

Use the `mget` command to test if an average has crossed a defined threshold.

For threshold examples, see **Admin > Alerting > Threshold Alerts**.

Use the following syntax:

```
mget last{duration} total|avg|nonzero  
[above|below {value}[%]] [time {time filter}]  
{type} {parent regex} {child regex} {attribute regex}  
[profile {profile name}] [any|all|not group {group name} ...]
```

### Examples

Retrieve a list of devices in a group whose average ping RTTs have exceeded a specific value in a specific timeframe:

```
mget last1h avg above 10000 rtt * /ping/ * any group data-center2
```

Retrieve a list of interfaces in a group whose usage have exceeded a specific percentage in a specific timeframe:

```
mget last5m avg above 80% * * * /ifInUtil|ifOutUtil/ any group wan-links3
```

Retrieve a list of devices whose CPU usage have exceeded a specific percentage in a specific timeframe:

```
mget last5m avg above 60 * * * /CISCO-PROCESS-MIB.cpmCPU  
Totalmin/
```

### 4.4.4 series

Use the `series` command to return time-series values for a selected time range.

Use the following syntax:

```
series [interval total|avg {secs}] time {time filter}  
{type} {parent regex} {child regex} {attribute regex}  
[profile {profile name}] [any|all|not group {group name} ...]
```

### Example

Retrieve the total In/OutOctet values for an interface:

```
series interval total 3600 time yesterday counter swt250  
Fa0/1|InOctets|OutOctets/
```

```
swt250 Fa0/1 IF-MIB.ifHCInOctets = 2524672,2463360,2449920,  
2447488,2486656,2488064,2457856,...  
swt250 Fa0/1 IF-MIB.ifHCOctets = 975552,806976,803392,  
822016,802496,894144,820480,820608,...
```

### 4.4.5 top

Use the `top` command to retrieve information on the highest number of events of a particular type within a selected time range.

Use the following syntax:

```
top {N} [reverse] [interval avg {secs}] total|max|avg|median {NN} time {time filter}
{type} {parent regex} {child regex} {attribute regex}
[profile {profile name}] [any|all|not group {group name} ...]
```

### Example

Retrieve the interfaces with the highest In/OutOctets for yesterday:

```
top 20 total time yesterday counter * * /IF-MIB.if.*Octets/
cisco-74-1-12 Tel1/1 IF-MIB.ifHCInOctets = 134374989234176
cisco-74-1-16 Tel1/1/1 IF-MIB.ifHCOutOctets = 92564298924032
juniper-74-2-7 xe-0/0/10.0 IF-MIB.ifHCInOctets = 92329710977024 juniper-74-2-
12 ae47.0 IF-MIB.ifHCOutOctets = 92308460535808
cisco-74-1-24 Ethernet1/23 IF-MIB.ifHCOutOctets = 89412016799744
cisco-74-1-24 Ethernet1/24 IF-MIB.ifHCOutOctets = 88919019356160 juniper-74-
2-10 reth0.2213 IF-MIB.ifHCOutOctets = 88832066191360 arista-74-0-40
Ethernet5/6 IF-MIB.ifHCInOctets = 88685106167808
cisco-74-1-17 Po55.152 IF-MIB.ifHCOutOctets = 88431959736320
...
```

## 4.5 Web API

Instead of using the command console (see 2.2) or SSH, you can transport commands via HTTP.

Use the following syntax:

```
http://{server}/api-db?password={pw};cmds={query}
```

The `api-rw` user can run any API command.

The `api-ro` user can use any of the following commands:

```
Aggregate
  calc
  check
  count cseries
  get (url encode the query, e.g. replace each space with +)
  list
  mcalc
  mget
  mgroup
  mlist
  mtime
  mtype
  netmask
  rget
  series tf tget top tz
```

## CONFIG AND EVENTS

### To activate the config and events web API:

Go to **Admin > API > Web API Settings**.

Click the **Config and Events** option **On**.

Click **Save**.

**AKiPS** Dashboards Reports Tools Admin New PDF Licensed to demo1 v21.7.1 User: admin

### Web API Settings

Availability ☐ Off Default off

**Config and Events** ☒ On Default off

Config Viewer ☒ On Default off

HTTP Log ☐ Off Default off

NetFlow ☐ Off Default off

NetFlow Time-series ☐ Off Default off

Site Script Functions ☒ On Default off

Switch Port Mapper ☒ On Default off

Syslog and Traps ☒ On Default off

Unused Interfaces ☒ On Default off

**Save**

### Availability

This is a Web API wrapper around the nm-availability program. Refer to the AKiPS programming guide.

Access: Requires a username of *api-ro*

Syntax: `http://{server}/api-availability?password={pw};{option}={value};...`

Options:

Option	Value	Comment
maintenance	on   off	Show/hide maintenance mode devices
mode	group   device   events	One only
time	{time filter}	Refer to the programming guide
report	ping4, ping6, snmp, ifstatus	Any combination, comma separated
entity	{device} {{child}}	
group	{group_name}	
profile	{profile_name}	

Example: `http://{server}/api-availability?password={pw};mode=group;time=last1d;report=ping4`

Sample Output:

```
ping4,PING.icmpState,Cisco,817150,815400,10000,last1d
ping4,PING.icmpState,FreeBSD,5832,5832,9500,last1d
```

### Config and Events

This is a Web API wrapper around the nm-db program. Refer to the *API Programming Guide*.

Access: Requires a username of *api-ro* and/or *api-rw*.

Username	Permissions
api-ro	Read-only commands
api-rw	All commands

Graphic 50. Activating the config and events web API

Go to **Admin > Users / Profiles > User Settings**.

In the **Username** text field, type `api-rw`

Complete the remaining text fields.

Click **Add**.

## CONFIG AND EVENTS

AKIPS Dashboards Reports Tools Admin New PDF Licensed to demo1 v21.7.1 User: admin

### User Settings

Username  Full Name  Password  ☒ Allow password changes Email  Profile

Username	Full Name	Auth	Email	Profile	Mute Time	
admin		local			1h 8h 1d 7d forever	Edit
api-ro	api-ro	local			1h 8h 1d 7d forever	Edit Delete
api-rw	api-rw	local			1h 8h 1d 7d forever	Edit Delete
njc-test	Neil-Test	local		Profile-Asia-Pac	1h 8h 1d 7d forever	Edit Delete
njf	Nick	local	nick@example.com	Profile-NetOps	1h 8h 1d 7d forever	Edit Delete
temp-user	Temp User	local	temp-user@akips.com	Helpdesk	1h 8h 1d 7d forever	Edit Delete

Graphic 51 Configuring user settings for the config and events web API

### Case study

A customer wanted to return data from the web API config and events endpoint in JSON. He used site scripting to loop through a list of devices and push the output to an array:

```
push(@deviceoutput, {devicename => $dev, ip4addr =>
$ip4addr, model => $model});
```

He then used the JSON library to convert the array to JSON format:

```
use JSON; my $jsonoutput = to_json(\@deviceoutput,
{utf8 => 1, pretty => 1, canonical => 1});
```

### 4.5.1 cURL

You can use cURL to access AKIPS data through the web API. Command line options include:

Command line	Description
<code>-k</code>	insecure mode; suppresses SSC error messages
<code>-d, --data</code>	sends POST data
<code>--compressed</code>	notifies the HTTP server to compress output
<code>--data-binary</code>	sends POST data in binary form: does not strip new line characters (e.g. gzip)
<code>@{filename}</code> <code>-d, --data</code>	reads POST data from {filename}. You must first use
<code>@-</code> <code>-d, --data</code>	reads POST data from standard input. You must first use

## Examples

Retrieve a list of devices:

```
curl "http://{server}/api-db?password={pw};cmds=mget+device+*"
```

Retrieve a list of unreachable IPv4 ping devices:

```
curl "http://{server}/api-db?password={pw};  
cmds=mget+*+*+ping4+PING.icmpState+value+/down/"
```

### 4.5.2 data

Use the `data` command to send data, including multiple commands at a time.  
You can run several commands from a file, e.g.

```
curl --data-binary @commands "http://{server}/api-db?password={pw}"
```

You can run several commands from the standard input stream, e.g.

```
cat commands | curl --data-binary @- "http://{server}/api-db?password={pw}"
```

## Examples

Retrieve a list of devices:

```
curl -d "cmds=mget device *" "http://{server}/api-db?password={pw}"
```

Retrieve a list of unreachable IPv4 ping devices:

```
curl -d "cmds=mget * * ping4 PING.icmpState value /down/"  
"http://{server}/api-db?password={pw}"
```

## 5 Syslog and traps reporter

AKIPS stores all syslog and SNMP traps in a single database.

Use the `nm-msg-reporter` command line tool to extract and filter messages.

Use the following syntax:

```
time {filter}
[type {syslog|trap}]
[addr {filter}]
[limit {num}] [
regex {filter}]
[mute {file}]
[trim {file}]
[opt stats]
[interval {sec}]
```

### 5.1 CSV output

Each syslog or trap message contains:

- header line: {system timestamp} {type} {IP version}  
{IP Address}
- message text
- blank terminating line.

### Examples

Retrieve all syslog messages for the past hour:

```
time last1h type syslog

1436229532 syslog 4 10.4.2.130
error local7 67: *Mar 3 09:22:52.269: SYS-3-CPUHOG: Task is running for
(3180)msecs, more ...

1436229532 syslog 4 10.4.2.130
error local7 68: -Traceback= 0x60C35A94 0x60C35C84 0x60C34688 0x60C34B84
0x60C38BC8 0x60C2950C ...

1436229532 syslog 4 10.4.2.130
error local7 69: *Mar 3 09:22:53.109: SYS-3-CPUHOG: Task is running for
(4024)msecs, more ...

1436229532 syslog 4 10.4.2.130
error local7 70: -Traceback= 0x60C35A98 0x60C35C84 0x60C34688 0x60C34B84
0x60C38BC8 0x60C2950C ...
```

## ***SYSLOG AND TRAPS REPORTER***

Retrieve all trap messages for the past day which contain specific text:

```
type trap time lastId regex OSPF 1436232075 trap 4 10.4.2.26
SNMPv2-MIB sysUpTime 0 TimeTicks 53803 SNMPv2-MIB snmpTrapOID 0
ObjectIdentifier CISCO-SYSLOG-MIB.clog MessageGenerated
CISCO-SYSLOG-MIB clogHistFacility 122 DisplayString OSPFv3 CISCO-SYSLOG-MIB
clogHistSeverity 122 ENUM 6,notice
CISCO-SYSLOG-MIB clogHistMsgName 122 DisplayString ADJCHG
CISCO-SYSLOG-MIB clogHistMsgText 122 DisplayString Process 1, Nbr
10.4.45.1 on Serial1/6 from ...
CISCO-SYSLOG-MIB clogHistTimestamp 122 TimeTicks 53803
```

```
1436232075trap 4 10.4.2.26
SNMPv2-MIB sysUpTime 0 TimeTicks 53803 SNMPv2-MIB snmpTrapOID 0
ObjectIdentifier OSPF-TRAP-MIB.ospf NbrStateChange
OSPF-MIB ospfRouterId 10.4.2.20 IPAddress 10.4.40.1
OSPF-MIB ospfNbrIpAddr 10.4.2.20 IPAddress 10.4.2.166
OSPF-MIB ospfNbrAddressLessIndex 10.4.2.20 Integer 0
OSPF-MIB ospfNbrRtrId 10.4.2.20 IPAddress 10.4.45.1 OSPF-MIB ospfNbrState
10.4.2.20 ENUM 1,down
```

Retrieve all syslog and trap messages for today from a specific IP address:

```
time today addr 10.4.2.26
1436232275 syslog 4 10.4.2.26
notice local7 149:Jul 7 11:24:34.476: LINEPROTO-5-UPDOWN: Line
protocol on Interface Serial1/6, changed...

1436232275 syslog 4 10.4.2.26
notice local7 150:Jul 7 11:24:34.572: OSPF-5-ADJCHG: Process 1, Nbr 10.4.45.1
onSerial1/6 from LOADING...

1436232275 trap 4 10.4.2.26
SNMPv2-MIB sysUpTime 0 TimeTicks 54003 SNMPv2-MIB snmpTrapOID 0
ObjectIdentifier OSPF-TRAP-MIB.ospf Nbr StateChange
OSPF-MIB ospfRouterId 10.4.2.20 IPAddress 10.4.40.1
OSPF-MIB ospfNbrIpAddr 10.4.2.20 IPAddress 10.4.2.166
OSPF-MIB ospfNbrAddressLessIndex 10.4.2.20 Integer 0
OSPF-MIB ospfNbrRtrId 10.4.2.20 IPAddress 10.4.45.1 OSPF-MIB ospfNbrState
10.4.2.20 ENUM 8,full

1436232276 trap 4 10.4.2.26
SNMPv2-MIB sysUpTime 0 TimeTicks 54004 SNMPv2-MIB snmpTrapOID 0
ObjectIdentifier OSPF-TRAP-MIB. ospf OriginateLsa
OSPF-MIB ospfRouterId 10.4.2.20 IPAddress 10.4.40.1
OSPF-MIB ospfLsdbAreaId 10.4.2.20 IPAddress 0.0.0.0 OSPF-MIB ospfLsdbType
10.4.2.20 ENUM 1,routerLink
```

## 5.2 Web API

You can use a web API wrapper for nm-msg-reporter

Use the following syntax:

```
https://{server}/api-msg?password={pw};time={time filter}; [addr={ip
filter}]; [type=syslog|trap]; [device={name}|{regex}]; [regex={regex
filter}]; [limit={qty messages}]
```

### Examples

Retrieve all syslog messages for the past 30 minutes for devices with specific text:

```
http://{server}/api-msg?password={pw};time=last30m;type=syslog;
device=/^swt/;
```

Retrieve up to a specific number of syslog messages for the past hour for a specific IP address:

```
http://{server}/api-msg?password={pw};time=last1h;addr=0.1.9.6;
type=syslog;regex=down;limit=5;
```

### To activate the syslog and traps web API:

Go to **Admin > API > Web API Settings**.

Click the **Syslog and Traps** option **On**.

Click **Save**.

**AKIPS** Dashboards Reports Tools Admin **New** PDF Licensed to demo1 v21.7.1 User: admin

#### Web API Settings

Availability ☐ Off Default off

Config and Events ☐ On Default off

Config Viewer ☐ On Default off

HTTP Log ☐ Off Default off

NetFlow ☐ Off Default off

NetFlow Time-series ☐ Off Default off

Site Script Functions ☐ On Default off

Switch Port Mapper ☐ On Default off

**Syslog and Traps ☒ On Default off**

Unused Interfaces ☐ On Default off

**Save**

#### Availability

This is a Web API wrapper around the nm-availability program. Refer to the AKIPS programming guide.

Access:  
Requires a username of *api-ro*

Syntax:  
`http://{server}/api-availability?password={pw};(option)=(value);...`

Options:

Option	Value	Comment
maintenance	on   off	Show/hide maintenance mode devices
mode	group   device   events	One only
time	{time filter}	Refer to the programming guide
report	ping4, ping6, snmp, ifstatus	Any combination, comma separated
entity	{device} [{child}]	
group	{group_name}	
profile	{profile_name}	

Example:  
`http://{server}/api-availability?password={pw};mode=group;time=last1d;report=ping4`

Sample Output:

```
ping4,PING.icmpState,Cisco,817150,815400,10000,last1d
ping4,PING.icmpState,FreeBSD,5832,5832,9500,last1d
```

#### Config and Events

This is a Web API wrapper around the nm-db program. Refer to the *API Programming Guide*.

Access:  
Requires a username of *api-ro* and/or *api-rw*.

Username	Permissions
api-ro	Read-only commands
api-rw	All commands

Graphic 52. Activating the syslog and traps web API



Go to **Admin > Users / Profiles > User Settings**.

In the **Username** text field, type `api-ro`

Complete the remaining text fields.

Click **Add**.

AKiPS

[Dashboards](#)
[Reports](#)
[Tools](#)
[Admin](#)

New

PDF

Licensed to demo1 v21.7.1 User: admin

User Settings

Username

api-ro

Full Name

Full Name

Password

Password

☒ Allow password changes

Email

Email Address

Profile

Profile Group

Add

Username	Full Name	Auth	Email	Profile	Mute Time	
admin		local			1h 8h 1d 7d forever	Edit
api-ro	api-ro	local			1h 8h 1d 7d forever	Edit Delete
api-rw	api-rw	local			1h 8h 1d 7d forever	Edit Delete
njc-test	Neil-Test	local		Profile-Asia-Pac	1h 8h 1d 7d forever	Edit Delete
njf	Nick	local	nick@example.com	Profile-NetOps	1h 8h 1d 7d forever	Edit Delete
temp-user	Temp-User	local	temp-user@akips.com	Helpdesk	1h 8h 1d 7d forever	Edit Delete

*G.53 Configuring user settings for the syslog and traps web API*

## 6 Availability

Use the `nm-availability` command line tool to retrieve availability statistics. Use the following syntax:

```
[-m] (include maintenance mode)
mode { group | device | events }
time { time filter }
report { ping4,ping6,snmp,ifstatus }
[entity { device } [{ interface }]]
[group { name }]
[profile { name }]
```

### 6.1 Group

This will retrieve a summary of device and interface groups.

The output will be in the following format:

```
{child},{attr},{group name},{total time},{match time},
{group target},{tf} [{group tf}]
```

#### Example

```
nm-availability mode group time last1w report ping4
```

```
ping4,PING.icmpState,1-Building-4,11688115,11687711,9990,last1w
ping4,PING.icmpState,1-Fraser,8213270,8213190,9990,last1w
ping4,PING.icmpState,1-Building-16,44541195,44540002,9990,last1w
ping4,PING.icmpState,Accedian,1766635,1766635,9890,last1w;
mon to sat 6:00 to 20:00
ping4,PING.icmpState,Aerohive,589475,589475,9999,last1w; mon to fri 7:00 to
19:00; sat 8:00 to 18:00
```

### 6.2 Device

This will retrieve a summary of devices. The output will be in the following format:

```
{parent},{child},{attr},{total time},{match time},{group target}
```

#### Example

```
nm-availability mode device time last1w report snmp,ping4
group Accedian
```

```
accedian-131-2-7,ping4,PING.icmpState,136020,136020,9890 accedian-131-2-
7,sys,SNMP.snmpState,136020,136020,9890 accedian-131-2-
8,ping4,PING.icmpState,136020,136020,9890 accedian-131-2-
8,sys,SNMP.snmpState,136020,136020,9890 accedian-131-2-
9,ping4,PING.icmpState,136020,136020,9890 accedian-131-2-
9,sys,SNMP.snmpState,136020,136020,9890
```

## 6.3 Events

This will retrieve pairs of up/down events. The output will be in the following format:

```
{parent},{child},{down},{up},{total time},{match_time}
```

### Example

```
nm-availability mode events time last1M report ping4  
entity cisco-131-16-1
```

```
cisco-131-16-1,ping4,1603822871,1603822916,2389764,2388341  
cisco-131-16-1,ping4,1603088563,1603089823,2389764,2388341  
cisco-131-16-1,ping4,1603060380,1603060498,2389764,2388341
```

## 7 NetFlow reporter

AKIPS stores NetFlow records in a dedicated database.

Use the `nm-flow-reporter` command line tool to query the database.

Use the following syntax:

```
nm-flow-reporter [optional parameters] time {time filter}  
exporter {exporter IP address}
```

Optional parameters:

```
[src_ip | dst_ip | any_ip | both_ip {IP filter}]  
  
[src_as | dst_as | any_as | both_as {AS Number|Name|Regex}]  
[src_idx | dst_idx | any_idx | both_idx {ifIndex}]  
[show src_ip,dst_ip,src_host,dst_host,src_idx,dst_idx,src_as,  
dst_as,geo,tos,proto,pkt,oct,flow,conv,tsf,url]  
  
[sort src_ip | dst_ip | src_idx | dst_idx | src_as | dst_as |  
pkt | oct | flow | conv]  
  
[proto {protocol.service}]  
[limit {num}]  
[sort_dir {f | r}]
```

## NETFLOW REPORTER

Parameter	Description
<code>src_ip</code>	source IPv4/6 address
<code>dst_ip</code>	destination IPv4/6 address
<code>src_idx</code>	SNMP index of source (input) interface
<code>dst_idx</code>	SNMP index of destination (output) interface
<code>src_host</code>	source hostname
<code>dst_host</code>	destination hostname
<code>src_as</code>	source AS number
<code>dst_as</code>	destination AS number
<code>geo</code>	location (country code) of IP addresses
<code>tos</code>	IP type of service
<code>proto</code>	IP protocol type
<code>pkt</code>	number of packets
<code>oct</code>	octets (bytes)
<code>flow</code>	flows
<code>conv</code>	number of conversations
<code>tsf</code>	filter that can be passed to <code>nm-flow-timeseries</code>

### Example

Retrieve TopN protocol data for packets/octets, sorted by octets:

```
http://{server}/api-flow?password={pw};exporter=10.0.0.254;  
time=last10m;show=proto,pkt,oct;sort=oct
```

## 7.1 CSV output

The first line of output contains 24 fields which describe the data.

Each following line contains data which match the parameters.

Empty fields represent an unreported value.

```
source IP,destination IP,source index,destination index,
source hostname,destination hostname,source AS number,
destination AS number,source AS name,destination AS name,
source location,destination location,type of service,protocol,
packets,octets,flows,conversations,timeseries filter,url,
milliseconds to process,records processed,records matched,
records included in result
```

### Example

```
nm-flow-reporter exporter 10.117.0.35 time last30m limit 10show
src_ip,dst_ip,proto,geo,tos,pkt,oct,flow sort pktsort_dir f
src_ip 188.24.60.104 #Src IP,Dst IP,,,,,,Src Location,Dst
Location,TOS Number,Protocol, Packets,Bytes,Flows,,,3,44346,221,213

188.24.60.104,213.138.70.62,,,,,,RO,RU,0,tcp.re-mail-ck,124,
74576,3,,
188.24.60.104,203.45.91.4,,,,,,RO,AU,0,icmp.echoreply,73,38921,2,,
188.24.60.104,65.60.39.90,,,,,,RO,US,0,udp.epmap,64,38303,2,,
188.24.60.104,65.60.39.90,,,,,,RO,US,0,udp.unknown,63,33321,2,,
188.24.60.104,58.167.215.31,,,,,,RO,AU,0,tcp.http,49,35703,1,,
188.24.60.104,202.159.32.2,,,,,,RO,ID,200,udp.domain,49,29878,1,,
188.24.60.104,203.45.91.4,,,,,,RO,AU,0,udp.z3950,49,35131,1,,
188.24.60.104,123.100.150.72,,,,,,RO,AU,0,tcp.timeserver,49,
31455,1,,
188.24.60.104,81.205.200.183,,,,,,RO,NL,0,tcp.snpp,48,20521,1,,
188.24.60.104,65.60.39.90,,,,,,RO,US,0,tcp.netbios-ns,48,12403,1,,
```

## 7.2 Web API

You can use a web API wrapper for `nm-flow-reporter`

Use the following syntax:

```
http://{server}/api-flow?password={pw};{option}={value};...
```

Option	Value
<code>exporter</code>	<code>{exporter IP}</code>
<code>time</code>	<code>{increments of 5 minutes}</code>
<code>show</code> <code>oct, flow, conv</code>	<code>src_ip, dst_ip, src_host, dst_host, src_idx,</code> <code>dst_idx, src_as, dst_as, geo, tos, proto, pkt,</code>
<code>sort</code>	<code>src_ip   dst_ip   src_idx   dst_idx  </code> <code>src_as   dst_as   pkt   oct   flow   conv</code>
<code>proto</code>	<code>{protocol}.{service}</code>
<code>src_ip   dst_ip  </code>	<code>{ip filter} any_ip   both_ip</code>
<code>src_as   dst_as  </code>	<code>{Autonomous System Filter} any_as   both_as</code>
<code>src_idx   dst_idx  </code>	<code>{Index Filter Number} any_idx   both_idx</code>
<code>sort_dir</code>	<code>f   r</code>

## NETFLOW REPORTER

### To activate the NetFlow web API:

Go to **Admin > API > Web API Settings**.

Click the **NetFlow** option **On**.

Click **Save**.

**AKIPS** Dashboards Reports Tools Admin New PDF Licensed to demo1 v21.7.1 User: admin

### Web API Settings

Availability ☐ Off Default: off

Config and Events ☐ On Default: off

Config Viewer ☐ On Default: off

HTTP Log ☐ Off Default: off

**NetFlow ☐ On** Default: off

NetFlow Time-series ☐ Off Default: off

Site Script Functions ☐ On Default: off

Switch Port Mapper ☐ On Default: off

Syslog and Traps ☐ On Default: off

Unused Interfaces ☐ On Default: off

**Save**

### Availability

This is a Web API wrapper around the nm-availability program. Refer to the AKIPS programming guide.

Access:  
Requires a username of *api-ro*

Syntax:  
`http://{server}/api-availability?password=(pw);(option)=(value);...`

Options:

Option	Value	Comment
maintenance	on   off	Show/hide maintenance mode devices
mode	group   device   events	One only
time	{time filter}	Refer to the programming guide
report	ping4, ping6, snmp, ifstatus	Any combination, comma separated
entity	{device} [{child}]	
group	{group_name}	
profile	{profile_name}	

Example:  
`http://{server}/api-availability?password=(pw);mode=group;time=last1d;report=ping4`

Sample Output:

```
ping4,PING.icmpState,Cisco,817150,815400,10000,last1d
ping4,PING.icmpState,FreeBSD,5832,5832,9500,last1d
```

### Config and Events

This is a Web API wrapper around the nm-db program. Refer to the API Programming Guide.

Access:  
Requires a username of *api-ro* and/or *api-rw*.

Username	Permissions
api-ro	Read-only commands
api-rw	All commands

### G54. Activating the NetFlow web API

Go to **Admin > Users / Profiles > User Settings**.

In the **Username** text field, type `api-ro`

Complete the remaining text fields.

Click **Add**.

**AKIPS** Dashboards Reports Tools Admin New PDF Licensed to demo1 v21.7.1 User: admin

### User Settings

Username

Full Name

Password

☒ Allow password changes

Email

Profile

**Add**

Username	Full Name	Auth	Email	Profile	Mute Time	
admin		local			1h 8h 1d 7d forever	Edit
api-ro	api-ro	local			1h 8h 1d 7d forever	Edit Delete
api-rw	api-rw	local			1h 8h 1d 7d forever	Edit Delete
njc-test	Neil-Test	local		Profile-Asia-Pac	1h 8h 1d 7d forever	Edit Delete
njf	Nick	local	nick@example.com	Profile-NetOps	1h 8h 1d 7d forever	Edit Delete
temp-user	Temp-User	local	temp-user@akips.com	Helpdesk	1h 8h 1d 7d forever	Edit Delete

### G55. Configuring user settings for the NetFlow web API



## 8 NetFlow time series

Use the `nm-flow-timeseries` command line tool to extract time-series values for:

- packets
- bytes (octets)
- bits per second
- flows.

Use the following syntax:

```
nm-flow-timeseries [optional parameters] time {time filter}
exporter {exporter IP address} interval {minutes}
```

Optional parameters:

```
[src_ip | dst_ip | any_ip | both_ip {IP filter}]
[src_as | dst_as | any_as | both_as {AS Number|Name|Regex}]
[src_idx | dst_idx | any_idx | both_idx {ifIndex}]
tos {number} [proto {protocol.service}]
```

### 8.1 CSV output

Each query delivers four lines of CSV output that may contain the following:

- **source/destination IP**
- **protocol**
- **start time** (POSIX timestamp)
- **interval in seconds**
- **time-series values.** AKIPS calculates these by dividing the time span by the interval, e.g. today divided by 60 minutes produces 24 time-series values.

#### Examples

```
time last1h src_ip 157.187.62.203 interval 10 exporter
172.16.1.17
```

```
157.187.62.203,,,,,Pkts,1436243100,600,69970,1236,44698,45,38252,
33,21
157.187.62.203,,,,,Bytes,1436243100,600,98279249,134122,62765460,
4296,53683296,3352,2009
157.187.62.203,,,,,Bits 1436243100,600,786233992,1072976,502123680,
34368,429466368,26816,16072
157.187.62.203,,,,,Flows,1436243100,600,32,41,30,45,38,31,21
```

## NETFLOW TIME SERIES

```
time last1h src_ip 10.1.8.62 proto udp.snmp interval 5 exporter 10.4.2.22

10.1.8.62,,,udp.snmp,Pkts,1436244300,300,126,114,125,128,495,125,
128,106,131,128,121,0,52
10.1.8.62,,,udp.snmp,Bytes,1436244300,300,76605,68846,75615,77655,
120197,75738,77745,64000,79708,...
10.1.8.62,,,udp.snmp,Bits,1436244300,300,612840,550768,604920,
621240,961576,605904,621960,512000,...
10.1.8.62,,,udp.snmp,Flows,1436244300,300,49,51,50,50,140,50,50,
48,52,50,22,0,16
```

## 8.2 Web API

You can use a web API wrapper for nm-flow-timeseries Use the following syntax:

```
http://{server}/api-flow-timeseries?password={pw};
{option}={value};...
```

Option	Value
exporter	{exporter IP}
time	{time filter}
interval	{increments of 5 minutes}
proto	{protocol}.{service}
src_ip   dst_ip   any_ip   both_ip	{ip filter}
src_as   dst_as   any_as   both_as	{AS Number Name Regex}
src_idx   dst_idx   any_idx   both_idx	{ifIndex}
tos	{number}

### Example

Retrieve five-minute time-series values for all flow records on a specific exporter:

```
http://{server}/api-flow-timeseries?password={pw};
exporter=10.0.0.254;time=last4h;interval=5
```

### Activate the NetFlow time-series web API:

Go to **Admin > API > Web API Settings**.

Click the **NetFlow Time-series** option **On**. Click

**Save**.

## NETFLOW TIME SERIES

**AKiPS** Dashboards Reports Tools Admin **New** PDF Licensed to demo1 v21.7.1 User: admin

### Web API Settings

Availability ☐ Off Default off

Config and Events ☐ On Default off

Config Viewer ☐ On Default off

HTTP Log ☐ Off Default off

NetFlow ☐ On Default off

**NetFlow Time-series ☐ On Default off**

Site Script Functions ☐ On Default off

Switch Port Mapper ☐ On Default off

Syslog and Traps ☐ On Default off

Unused Interfaces ☐ On Default off

**Save**

### Availability

This is a Web API wrapper around the nm-availability program. Refer to the AKiPS programming guide.

Access:  
Requires a username of `api-ro`

Syntax:  
`http://{server}/api-availability?password={pw};{option}={value};...`

Options:

Option	Value	Comment
maintenance	on   off	Show/hide maintenance mode devices
mode	group   device   events	One only
time	{time filer}	Refer to the programming guide
report	ping4, ping6, snmp, ifstatus	Any combination, comma separated
entity	{device} [{child}]	
group	{group_name}	
profile	{profile_name}	

Example:  
`http://{server}/api-availability?password={pw};mode=group;time=lastId;report=ping4`

Sample Output:

```
ping4,PING.icmpState,Cisco,817150,815400,10000,lastId
ping4,PING.icmpState,FreeBSD,5832,5832,9500,lastId
```

### Config and Events

This is a Web API wrapper around the nm-db program. Refer to the API Programming Guide.

Access:  
Requires a username of `api-ro` and/or `api-rw`.

Username	Permissions
api-ro	Read-only commands
api-rw	All commands

### G56. Activating the NetFlow time series web API

Go to **Admin > Users / Profiles > User Settings**.

In the **Username** text field, type `api-ro`

Complete the remaining text fields.

Click **Add**.

**AKiPS** Dashboards Reports Tools Admin **New** PDF Licensed to demo1 v21.7.1 User: admin

### User Settings

**Username**

**Full Name**

**Password**

☒ Allow password changes

**Email**

**Profile**

**Add**

Username	Full Name	Auth	Email	Profile	Mute Time	
admin		local			1h 8h 1d 7d forever	Edit
api-ro	api-ro	local			1h 8h 1d 7d forever	Edit Delete
api-rw	api-rw	local			1h 8h 1d 7d forever	Edit Delete
njc-test	Neil-Test	local		Profile-Asia-Pac	1h 8h 1d 7d forever	Edit Delete
njf	Nick	local	nick@example.com	Profile-NetOps	1h 8h 1d 7d forever	Edit Delete
temp-user	Temp-User	local	temp-user@akips.com	Helpdesk	1h 8h 1d 7d forever	Edit Delete

### G57. Configuring user settings for the NetFlow time series web API

## 9 Switch port mapper

Switch port mapper data is stored in CSV format.

You can access it via:

- the command console (see 2.2)
- system log viewer (Go to Admin > System > System Log Viewer)
- web API (see 9.2).

### 9.1 CSV output

Filename	Description	Output format (CSV)
arp	ARP tables	<code>epoch,ipaddress,mac</code>
ip2mac	IP to MAC mapping	<code>epoch,mac,ipaddresses</code>
mac2ip	MAC to IP mapping	<code>epoch,mac,ipaddresses</code>
mac2sp	MAC to switch interface mapping	<code>epoch,mac,switch,port</code>
mac2vspa	MAC to vendor, switch, interface, IP address	<code>mac,vendor,switch,port,ipaddress</code>
sp2mac	switch interface to MAC	<code>switch,port,mac</code>
sp2vlan	switch interface to VLAN	<code>epoch,switch,port,vlan</code>
vlan2sp	VLAN to switch interface	<code>epoch,vlan,switch,port</code>
vlangs	list of all VLANs	<code>vlan</code>

## 9.2 Web API

### Activate the switch port mapper web API:

Go to **Admin > API > Web API Settings**.

Click the **Switch Port Mapper** option **On**.

Click **Save**.

**AKiPS** Dashboards Reports Tools Admin **New** PDF Licensed to demo1 v21.7.1 User: admin

### Web API Settings

Availability ☐ Off Default off

Config and Events ☐ On Default off

Config Viewer ☐ On Default off

HTTP Log ☐ Off Default off

NetFlow ☐ Off Default off

NetFlow Time-series ☐ Off Default off

Site Script Functions ☐ On Default off

**Switch Port Mapper ☒ On** Default off

Syslog and Traps ☐ On Default off

Unused Interfaces ☐ On Default off

**Save**

### Availability

This is a Web API wrapper around the nm-availability program.  
Refer to the AKiPS programming guide.

Access:  
Requires a username of *api-ro*

Syntax:  
`http://(server)/api-availability?password=(pw);(option)=(value);...`

Options:

Option	Value	Comment
maintenance	on   off	Show/hide maintenance mode devices
mode	group   device   events	One only
time	{time filer}	Refer to the programming guide
report	ping4, ping6, snmp, ifstatus	Any combination, comma separated
entity	{device} [{child}]	
group	{group_name}	
profile	{profile_name}	

Example:  
`http://(server)/api-availability?password=(pw);mode=group;time=last1d;report=ping4`

Sample Output:

```
ping4,PING icmpState,Cisco,817150,815400,10000,last1d
ping4,PING icmpState,FreeBSD,5832,5832,9500,last1d
```

### Config and Events

This is a Web API wrapper around the nm-db program.  
Refer to the API Programming Guide.

Access:  
Requires a username of *api-ro* and/or *api-rw*.

Username	Permissions
api-ro	Read-only commands
api-rw	All commands

*G58. Activating the switch port mapper web API*

Go to **Admin > Users / Profiles > User Settings**. In the **Username** text field, type `api-ro`

Complete the remaining text fields.

Click **Add**.

**AKiPS** Dashboards Reports Tools Admin **New** PDF Licensed to demo1 v21.7.1 User: admin

### User Settings

**Username**

**Full Name**

**Password**

☒ Allow password changes

**Email**

**Profile**

**Add**

Username	Full Name	Auth	Email	Profile	Mute Time	
admin		local			1h 8h 1d 7d forever	Edit
api-ro	api-ro	local			1h 8h 1d 7d forever	Edit Delete
api-rw	api-rw	local			1h 8h 1d 7d forever	Edit Delete
njc-test	Neil-Test	local		Profile-Asia-Pac	1h 8h 1d 7d forever	Edit Delete
njf	Nick	local	nick@example.com	Profile-NetOps	1h 8h 1d 7d forever	Edit Delete
temp-user	Temp-User	local	temp-user@akips.com	Helpdesk	1h 8h 1d 7d forever	Edit Delete

*G59. Configuring user settings for the switch port mapper web API*

## 10 Unused interfaces

### 10.1 CSV output

Field	Description
1	device name
2	interface name
3	interface speed
4	used/free
5	current IF-MIB.ifOperStatus
6	last change time (epoch timestamp (seconds))
7	IF-MIB.ifAlias
8	VLAN name

### 10.2 Web API

#### To activate the unused interfaces web API:

Go to **Admin > API > Web API Settings**. Click the **Unused Interfaces** option **On**.

Click **Save**.

## UNUSED INTERFACES

**AKiPS** Dashboards Reports Tools Admin **New** PDF Licensed to demo1 v21.7.1 User: admin

### Web API Settings

Availability ☐ Off Default off

Config and Events ☐ On Default off

Config Viewer ☐ On Default off

HTTP Log ☐ Off Default off

NetFlow ☐ Off Default off

NetFlow Time-series ☐ Off Default off

Site Script Functions ☐ On Default off

Switch Port Mapper ☐ On Default off

Syslog and Traps ☐ On Default off

**Unused Interfaces ☒ On** Default off

**Save**

### Availability

This is a Web API wrapper around the nm-availability program. Refer to the AKiPS programming guide.

Access:  
Requires a username of api-ro

Syntax:  
`http://(server)/api-availability?password=(pw);(option)=(value);...`

Options:

Option	Value	Comment
maintenance	on   off	Show/hide maintenance mode devices
mode	group   device   events	One only
time	(time filter)	Refer to the programming guide
report	ping4, ping6, snmp, ifstatus	Any combination, comma separated
entity	{device} [{child}]	
group	{group_name}	
profile	{profile_name}	

Example:  
`http://(server)/api-availability?password=(pw);mode=group;time=last1d;report=ping4`

Sample Output:

```
ping4,PING icmpState,Cisco,817150,815400,10000,last1d
ping4,PING icmpState,FreeBSD,5832,5832,9500,last1d
```

### Config and Events

This is a Web API wrapper around the nm-db program. Refer to the API Programming Guide.

Access:  
Requires a username of api-ro and/or api-rw.

Username	Permissions
api-ro	Read-only commands
api-rw	All commands

### G60. Activating the unused interfaces web API

Go to **Admin > Users / Profiles > User Settings**.

In the **Username** text field, type `api-ro`

Complete the remaining text fields.

Click **Add**.

**AKiPS** Dashboards Reports Tools Admin **New** PDF Licensed to demo1 v21.7.1 User: admin

### User Settings

Username  api-ro

Full Name  Full Name

Password  Password

☒ Allow password changes

Email  Email Address

Profile  Profile Group

**Add**

Username	Full Name	Auth	Email	Profile	Mute Time	
admin		local			1h 8h 1d 7d forever	Edit
api-ro	api-ro	local			1h 8h 1d 7d forever	Edit Delete
api-rw	api-rw	local			1h 8h 1d 7d forever	Edit Delete
njc-test	Neli-Test	local		Profile-Asia-Pac	1h 8h 1d 7d forever	Edit Delete
njf	Nick	local	nick@example.com	Profile-NetOps	1h 8h 1d 7d forever	Edit Delete
temp-user	Temp-User	local	temp-user@akips.com	Helpdesk	1h 8h 1d 7d forever	Edit Delete

### G61. Configuring user settings for the unused interfaces web API

## 11 TCP socket

You can use the TCP socket daemon to remotely access `nm-db` and `nm-msg-reporter`

It does not perform authentication, so use it only in a secure environment.

Use the following syntax:

```
{program} {socket number} [{Restrict IP Address}]
```

### Examples

Start an `nm-db` process listening on a port:

```
nm-db 3000
```

Start an `nm-db` process listening on a port and allow connections from a specific IP address:

```
nm-db 3001 10.0.0.50
```

Start an `nm-msg-reporter` process listening on a port and allow connections from a specific IP address:

```
nm-msg-reporter 3002 10.0.0.51
```



## 12 Perl modules

### 12.1 Common

This module is located in /usr/local/akips/pm/Akips/Common.pm

Review the module for available definitions and externally visible functions.

#### 12.1.1 Useful arrays

```
@weekday_names_short = (  
"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"  
);  
  
@weekday_names_long = (  
"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday",  
"Sunday"  
);  
  
@month_names_short = (  
"Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov",  
"Dec"  
);  
  
@month_names_long = (  
"January", "February", "March", "April", "May", "June",  
"July",  
"August", "September", "October", "November", "December"  
);  
  
@days_of_month = (  
"1st", "2nd", "3rd", "4th", "5th", "6th", "7th", "8th", "9th",  
"10th", "11th", "12th", "13th", "14th", "15th", "16th",  
"17th", "18th", "19th", "20th", "21st", "22nd", "23rd",  
"24th", "25th", "26th", "27th", "28th", "29th", "30th", "31st"  
);  
  
@days_in_month = (  
31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31  
);  
  
@metric_prefix = (  
'p', 'n', 'u', 'm', '', 'K', 'M', 'G', 'T', 'P', 'E'  
);
```

### 12.1.2 PRINT\_LINE

`PRINT_LINE` is useful when debugging. It prints the following on a single line to stderr:

- date/time
- process name
- function name
- filename
- line number
- error message.

#### Example

```
Jan 6 13:20:51.583957 0 ms nm-site-script-alert_uplink 44232
Akips::Site::alert_uplink Site.pm:48
Service returned: "401 Unauthorized"
```

### 12.1.3 trim

`trim` strips white space (spaces, line breaks, etc) from a string.

#### Example

```
my $str = " Hello World ! ";
$str = trim ($str);
print ($str); # "Hello World!"
```

### 12.1.4 errlog

AKIPS uses its own error log. Use the following syntax:

```
errlog ({LOG_LEVEL}, {log message});
```

Available log levels:

```
$ERR_FATAL
```

```
$ERR_ERROR
```

```
$ERR_WARNING
```

```
$ERR_INFO
```

```
$ERR_USER
```

```
$ERR_DEBUG
```

```
$ERR_CGI
```

**Example**

```
my $error_msg = "it broke";
errlog ($ERR_FATAL, "something went wrong - %s",
$error_msg);
```

**To view error messages:**

Go to **Admin > System > System Log Viewer**.

**12.1.5 get\_localtime()**

get\_localtime() (epoch timestamp) is a wrapper around the standard localtime()

Element	Description
<code>\$hash{sec}</code>	seconds: 00 to 59
<code>\$hash{min}</code>	minutes: 00 to 59
<code>\$hash{hour}</code>	hours: 00 to 23
<code>\$hash{wday}</code>	day of the week: 0 = Sunday to 6 = Saturday
<code>\$hash{mday}</code>	day of the month: 1 to 31
<code>\$hash{mon}</code>	month: 0 = January to 11 = December
<code>\$hash{year}</code>	year, e.g. 2020
<code>\$hash{yday}</code>	day of the year: 0 to 364 (0 to 365 in leap years)
<code>\$hash{isdst}</code>	daylight saving time: 0 = No, 1 = Yes

**Example**

```
my $h_ref = get_localtime (); printf STDOUT "%s %s %s",
$days_of_month[$h_ref->{mday} - 1],
$month_names_long[$h_ref->{mon}],
$h_ref->{year};
```

### 12.1.6 mail()

`mail()` takes a hash reference containing:

- to
- subject
- body.

#### Example

```
my @body =
    ( "Hello,",
      "Greetings from AKIPS!",
      "Sincerely,",
      "Bob"
    );

@files = ("/path/to/graph1.png", "/path/to/report1.pdf",
          "/path/to/report2.pdf");

mail ({
    to      => 'admin@example.com',
    subject => 'Greetings',
    body    => \@body,
    attach  => \@files, #(optional)
});
```

### 12.1.7 syslog()

Element	Description	Options
<code>ipaddr</code>	destination IP address	
<code>priority</code>	default: notice	emergency   alert   critical   error   warning   notice   info   debug
<code>facility</code>	default: user	auth   authpriv   console   cron   daemon   ftp   kern   lpr   mail   news   ntp   security   syslog   user   uucp   local0   local1   local2   local3   local4   local5   local6   local7
<code>message</code>	message body	

**Example**

```
syslog ({
    ipaddr = > "10.50.1.100",
    priority = > "error",
    facility = > "local3",
    message = > "The quick brown fox has jumped",
});
```

**12.1.8 http\_send() and http\_result()**

These HTTP functions are a wrapper around the `nm-http` command line tool.

`http_send()` sends an HTTP request.

`http_result()` sends an HTTP request and returns response data.

Element	Value	Comment
<code>url</code>	<code>http://www.example.com</code>	HTTP
<code>method</code>	GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, TRACE, PATCH	case insensitive
<code>content_type</code>	<code>text/html</code>	application/x-www- form- urlencoded = default
<code>headers</code>		send custom HTTP headers
<code>proxy</code>	<code>host:port</code>	send request through the specified proxy
<code>secure</code>	0 or 1	0 = allow SSC 1 = default
<code>show_headers</code>	0 or 1	0 = default 1 = show headers
<code>data</code>	POST data	

## 12.2 AKIPS database

The command line tool for the ADB is `nm-db`.

The ADB module simplifies interacting with `nm-db` by automatically opening and closing a bidirectional pipe when loading and exiting.

### 12.2.1 `adb_send()`

Use `adb_send()` to write the specified command to `nm-db`.

#### Example

```
adb_send ("add device group CoreRouters");  
adb_send ("add device group CoreSwitches"); adb_flush ();
```

### 12.2.2 `adb_flush()`

Use `adb_flush()` to flush all buffered output to `nm-db`.

You must do this after calling `adb_send()`

### 12.2.3 `adb_result()`

Use `adb_result()` to write to `nm-db` and return the results in either a scalar or array.

You do not need to call `adb_flush()`

## 12.3 Discover

### 12.3.1 `discover_scan()`

`discover_scan()` performs the first stage of the device discover and creates the intermediate files for `discover_config()`

`ping.scan` lists the IPv4/6 addresses that responded to a ping scan.

`snmp.scan` lists the SNMP walk of the SNMPv2-MIB.system for each device.

`devices` lists the IPv4/6 address, SNMPv2-MIB.sysName, SNMP credentials, SNMPv2-MIB.sysObjectID and SNMPv2-MIB.sysDescr.

Use the following syntax:

```
discover_scan ( {SNMP Parameters}, {IP Address Range}, ... );
```

## Examples

Scan three IP ranges using existing discover SNMP credentials:

```
discover_scan (undef, "10.0.0.0/16", "10.3.0.0/24", "10.5.0.0/24");
```

Scan multiple IP ranges using specified SNMPv2 credentials:

```
discover_scan ("version 2 community loofah", "10.3.0.0/24", "10.5.0.0/24");
```

Scan one IP address using specified SNMPv3 credentials:

```
discover_scan ("version 3 sha passwd1 aes256 passwd1", "10.2.0.1");
```

### 12.3.2 discover\_config()

`discover_config()` performs SNMP walks of each device which  
`discover_scan()` locates, then processes the data to configure each device.

### 12.3.3 discover\_device\_rewalk()

`discover_device_rewalk()` performs an SNMP walk for a specified device and then processes the data to configure the device.

It returns:

0 = failure

1 = success

Use the following syntax:

```
discover_device_rewalk (\%hash_ref);
```

## Example

```
discover_device_rewalk ({  
  ipaddr => "10.1.2.3",  
  device => "atlanta-ro",  
})
```

## 13 Site scripts

You can use site scripts to:

- configure AKIPS after discover
- configure AKIPS after auto grouping
- schedule periodic scripting.

### Case study

A customer built a custom report in AKIPS by using fields in existing reports.

He used site scripting to load up arrays as `mget` outputs, added the necessary logic to align the data into hashes or whatever nested structure by whatever key fields, and then within a loop did a `printf` to produce a csv output.

He used smaller scripts with the `custom_ script` prefix to break them into more digestible and reusable subroutines, e.g. a function that only made DB calls.

### 13.1 Using site scripts

#### To use site scripts:

Go to the AKIPS website (<https://www.akips.com>).

Go to **Support > Site Scripts**.

#### To use an existing site script:

Copy the relevant script.

Open AKIPS and go to **Admin > API > Site Scripting**.

Paste the script into the text field.

#### To create a new site script:

Copy an existing AKIPS script which is similar to what you need.

Open AKIPS and go to **Admin > API > Site Scripting**.

Paste the script into the text field.

Modify the script.



## 24. Interface Speeds by Name

This script manually sets interface speeds by specifying *device name* and *interface name*.

Site Scripting uses function names starting with *ifspeed\_* to run commands to manually set interface speeds.

You can test your *ifspeed\_* code by selecting "All ifspeed\_ functions" from the dropdown, then clicking *Run*. Make sure there are no error messages in the output on the right side of the screen. You can then check an Interface report to confirm that the speed has been set correctly.

If you only need to manually set speeds on a handful of interfaces, you can issue individual *set* commands specifying the device and interface names. The following example sets the speed to 2Gbps for 3 interfaces on the device *atlanta-ro*.

Download script

```
sub ifspeed_interfaces
{
    # Command syntax:
    # set {device} {interface} IF-MIB.ifSpeed = {speed in bits per sec}

    adb_send ("set atlanta-ro Te3/1 IF-MIB.ifSpeed = 2000000000");
    adb_send ("set atlanta-ro Te3/2 IF-MIB.ifSpeed = 2000000000");
    adb_send ("set atlanta-ro Te3/3 IF-MIB.ifSpeed = 2000000000");
    adb_flush ();
}
```



## G62. Selecting and copying a site script

AKIPS
Dashboards
Reports
Tools
Admin
New
PDF

Licensed to demo1 - v21.7.1
User: admin

Site Scripting
All alert\_functions
Run
Save
Help

```

1388 {
1389     my $IN;
1390     my $OUT;
1391     my $snmp_cfg_ref = config_load_snmp_cfg ();
1392     my $snmp_pid = open2 ($IN, $OUT, $SNMP_PROG) or EXIT_FATAL ("Can't start snmp process: $!");
1393     my $domains = config_load_domains ();
1394     my $ip2dev;
1395     for my $line (adb_result ("mget text * sys SNMP.ipaddr")) {
1396         my ($device, undef, undef, undef, $ipaddr) = split (" ", $line, 5);
1397         $ip2dev{$ipaddr} = $device;
1398     }
1399     for my $ip (sort keys $ip2dev) {
1400         my $device = $ip2dev{$ip};
1401         my $snmp_param = $snmp_cfg_ref->{lc ($device)}{cmd} || "";
1402         printf $OUT "get %s %s SNMPv2-MIB.sysName.0\n", $ip, $snmp_param;
1403     }
1404     close $OUT;
1405     my @del_devices;
1406     while (my $line = <$IN>) {
1407         chomp $line;
1408         my ($ip, undef, undef, undef, $sysname) = split (" ", $line, 5);
1409         my $device = $ip2dev{$ip};
1410         next if (not defined $device);
1411         next if (not defined $sysname);
1412         $sysname = config_strip_sysname ($sysname, $domains);
1413         if ($device ne $sysname) {
1414             printf "Device name %s mismatch sysName %s\n", $device, $sysname;
1415             push @del_devices, $device;
1416         }
1417     }
1418     close $IN;
1419     #config_delete_device (@del_devices);
1420 }
1421 }
1422 }
1423 }
1424 }
1425 }
1426 }
```

NOTE: This is an "expert" only section. Assistance from AKIPS technical support may be required.

Site scripting is for performing:

- Post discover configuration
- Post auto grouping configuration
- Periodic scheduled scripting

Function names must start with the following prefixes, and will run at the times listed below.

Function name prefix	Schedule
alert_	Called by Alerting rules
config_	End of Discover/Rewalk
custom_	Manually via Site Scripting
group_	End of Discover/Rewalk or Auto Grouping
ifspeed_	End of Discover/Rewalk or Tune Interface Speed script
sched_Nm_	Every N minutes (e.g. sched_15m_do_stuff)
sched_Nh_	Every N hours (e.g. sched_1h_do_more_stuff)
sched_HHMM_	Run at a scheduled time (e.g. sched_0930_report)
web_	Remote http/https client

NOTES

- All functions are built into a single Perl module.
- Loading ensures only one instance of each type can run.
- Functions are run in order.

Alert Scripting

Alerting can use the `call` syntax to call a function in Site Scripting. The parameters passed through to the function vary depending on the type of alert (Status, Syslog, Threshold, or Trap).

The parameters are passed in via a hash reference, as per the following example:

```

sub alert_ifoperstatus_uplink
{
    my ($arg_ref) = @_;
    my $device = $arg_ref->{device};

    # ... code goes here ...
}
```

## G63. Pasting a site script into AKIPS

Click **Save**.

AKIPS will kill the script if it runs for longer than two minutes.

Refresh the page. The drop-down list will display functions which you can manually run from this page.

## 13.2 Naming a site script

When naming a script, use one of the following prefixes:

Prefix	Schedule
<code>admin_</code>	called by alerting rules
<code>config_</code>	end of discover/rewalk
<code>custom_</code>	manual execution
<code>group_</code>	end of discover/rewalk or auto grouping
<code>ifspeed_</code>	end of discover/rewalk or interface speed update
<code>sched_1m_</code>	every minute
<code>sched_5m_</code>	every five minutes
<code>sched_1h_</code>	every hour
<code>sched_HHMM_</code>	every day at a specific time
<code>web_</code>	triggered by remote HTTP client

After you save the script, AKIPS will check the syntax and notify you of any errors.

## 13.3 Alerting site scripts

Alerting site scripts supports the following alerts:

- status
- scheduled
- syslog
- threshold
- trap.

**To test an alerting site script:**

Go to **Admin > API > Site Scripting**.

## *SITE SCRIPTS*

Enter a hash table of test data for the subroutine you wish to test. Note the following rules:

### **Name**

The name of the hash table must be identical to the subroutine.

### **Key-value pairs**

Each alert type requires specific key-value pairs. Each pair must appear on a single line.

AKIPS will interpret undefined pairs as an empty string.

Pairs are delimited by = >

### **Arrays**

Arrays must open and close with [ ] (square brackets).

Arrays can be either single- or multi-line.

Multi-line arrays must have one value per line.

### **Values**

Values can be either scalar or array.

### **Closing punctuation**

The closing ); (parenthesis and semicolon) must be on their own line.

Click **Save**.

From the drop-down menu, select **All alert\_ functions**.

Click **Run**.

## **Examples**

### **Status\_**

```
our %alert_{foo}_status =(
    device => "device_value",
    child  => "child_value",
    attr   => "attr_value",
    state  => "state_value",
);

our %alert_mail_status = (
    device => "CISCO-82-1-109",
    child => "gill/1",
    attr=> "IF-MIB.ifOperStatus",
    state => "down",
);
```

## SITE SCRIPTS

### Syslog\_

```
our %alert_{foo}_syslog = (  
    device=> "device_value",  
    ipaddr=> "ipaddr_value",  
    msg=> "msg_value",  
);  
  
our %alert_mail_syslog = (  
    device=> "eaton-123-1-190",  
    ipaddr=> "10.82.0.109",  
    msg=> "161 20 1",  
);
```

### Trap\_

```
our %alert_{foo}_trap = (  
    device=> "device_value",  
    ipaddr=> "ipaddr_value", trap_oid => "  
trap_oid_value",  
    uptime=> "uptime_value",  
    oids=> [ "oid_1",  
            "oid_x",  
            "oid_n-1",  
            "oid_n" ],  
);  
  
our %alert_mail_trap = (  
    device=> "cisco-82-0-109",  
    ipaddr=> "10.82.0.109",  
    trap_oid => "EdgeSwitch-SWITCHING-MIB.  
fastPathSwitchingTraps.29",  
    uptime=> "1541842604",  
    oids=> [ "EdgeSwitch-SWITCHING-MIB.agentLoginSession Index: ",  
            "EdgeSwitch-SWITCHING-MIB.agentLoginSession UserName: akips, ",  
            "EdgeSwitch-SWITCHING-MIB.agentLoginSession ConnectionType: 3,ssh",  
            "EdgeSwitch-SWITCHING-MIB.agentLoginSessionInet Address:  
10.0.19.2",  
            "EdgeSwitch-SWITCHING-MIB.  
agentLoginSessionStatus:1,active" ],  
);
```