Pandora FMS Command Line Interface (CLI)

Pandora FMS Command Line Interface (CLI)

Pandora FMS CLI

The Pandora FMS CLI (Command-Line Interface) is used for making calls in command line on the file `pandora_manage`.

This method is specially useful to integrate applications of thirds parts with Pandora FMS through automated tasks. Basically, it consists on one call with the parameters formated to do and action such as the creation an elimination of one agent, one module or one user, among other things.

```
pandora_manage <pandora_server.conf path> <option> <option parameters>  [
optional parameters ]
```

With backslash \ as a line connector:

```
pandora_manage \
    <pandora_server.conf path> \
    <option> \
    <option parameters> \
    [ optional parameters ]
```

Syntax

- All parameters, mandatory or optional, can be delimited with single quotes. This is especially useful when a parameter has one or more spaces.
- If you need to omit a parameter you can use two single quotes together. In case the parameter is strictly necessary a message like `[ERROR] Error:` will be displayed followed by a description of the parameter. This is also useful to indicate empty optional parameters.
- Some parameters are case sensitive, others are case insensitive. For example, when creating an agent the name of the PFMS server must be strictly the same, however the name of the primary group to which the agent will belong can be in upper and/or lower case and if there is a match the group will be assigned.
- You can use the alias of an agent, instead of its name itself, by means of the `use_alias` parameter at the end of the command line.

Help

General help:

```
pandora_manage --h
```

To obtain help about one specific option just write option without parameters:

```
pandora_manage /etc/pandora/pandora_server.conf --create_user
```

Agents

create_agent

Required parameters (syntax):

- < agent name > Returns error if the name exists (see < alias as name >).
- < operating system >
- < group name > Must match an existing OS (case insensitive) otherwise it will be registered without any group.
- < server name > Be case sensitive, otherwise the value of the field will be left null *without any warning*. You must verify that the agent has been assigned to a server.

Optional parameters:

- < IP address or URL >
- < description > If you omit the description `Created by < server name >` will be added (regardless of whether the specified server exists or not).
- < interval > In seconds, default value: 300 (see next point).
- < alias as name > The default value is `1` and with this value the agent alias will be the same as the agent name. If you want the name to be randomly generated, use `0`.

Description:

An agent will be created with the specified name, operating system, group and server. Optionally, it can be given an address (IP address or name), a description and an interval in seconds.

If you are going to use a random agent name using `0` as the last parameter, the agent period must be specified, the recommended value being 300 seconds.

Example (see call and syntax):

An agent will be created with a random name and the alias `My agent` and with the recommended period of 300 seconds.

```
pandora_manage /etc/pandora/pandora_server.conf --create_agent \
```

```
    'My agent' AIX Databases Central-Server \
    192.168.12.123 'Agent description' '' 0
```

update_agent

Required parameters (syntax):

- < name_of_agent >
- < field_to_change >
- < new_value >

Optional parameters:

- use_alias

Description: An existing agent's < field_to_modify > will be updated. The available fields are the following:

- agent_name
- address
- description
- group_name
- interval
- os_name
- disabled (0 to activate it, 1 to inactivate it).
- parent_name
- cascade_protection (0 to activate it, 1 to inactivate it).
- icon_path
- update_gis_data (0 to activate it, 1 to inactivate it).
- custom_id

To identify the agent by its alias instead of its name, use use_alias right at the end.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --update_agent \
    'My Agent' agent_name 'Agent 2'
```

When changing an agent's IP address note that this command allows repeating IP addresses.

```
pandora_manage /etc/pandora/pandora_server.conf --update_agent \
    'My Agent' address 192.168.7.47
```

```
pandora_manage /etc/pandora/pandora_server.conf --update_agent \
    'My Agent' description 'New note' \
```

```
    use_alias
```

For the `disabled` option only `0` and `1` should be used, any other value will produce unpredictable search and/or display results in the Web Console.

```
pandora_manage /etc/pandora/pandora_server.conf --update_agent \
    'My Agent' disabled 1 \
    use_alias
```

For the `cascade_protection` option only `0` and `1` should be used, any other value will produce unpredictable search and/or display results in the Web Console.

```
pandora_manage /etc/pandora/pandora_server.conf --update_agent \
    'My Agent' cascade_protection 1 \
    use_alias
```

agent_update_custom_fields

Required parameters (syntax):

- < id_agent > agent identifier.
- < field_type > field type which can be 0 if it is a text type or 1 if it is a combo box (select) type composed of several options. See section on answers.
- < field_to_change > name of the field to be updated.
- < valor_nuevo > the value of the field to be updated.

Description: updates the custom fields of an agent.

The answers of which they are composed are:

- ERROR:
    - If the agent does not exist.
    - If the field does not exist.
    - In the event that the `new_value` of a combo does not match the ones you have preset.
    - If you were unable to update the field.
- INFO:
    - Updating the X field with the agent id Y.
    - Correctly updated.

Examples (see call and syntax):

TEXT TYPE:

```
pandora_manage /etc/pandora/pandora_server.conf \
 --agent_update_custom_fields \
 1 0 'Serial number' 'THX-1138'
```

COMBO TYPE: you must have configured a custom field named `Names` with at least one item named `John`.

```
pandora_manage /etc/pandora/pandora_server.conf \
 --agent_update_custom_fields \
 1 1 'Names' 'John'
```

delete_agent

Required parameters (syntax):

- < agent_name >

Optional parameter:

- `use_alias` It should be noted that agent aliases can be repeated and several agents can be deleted with a single command.

Description:

The agent(s) will be deleted by passing its name or alias as a parameter.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf \
    --delete_agent 'pandora.internals' \
    use_alias
```

disable_group

Manadory parameter:

- < group_name >

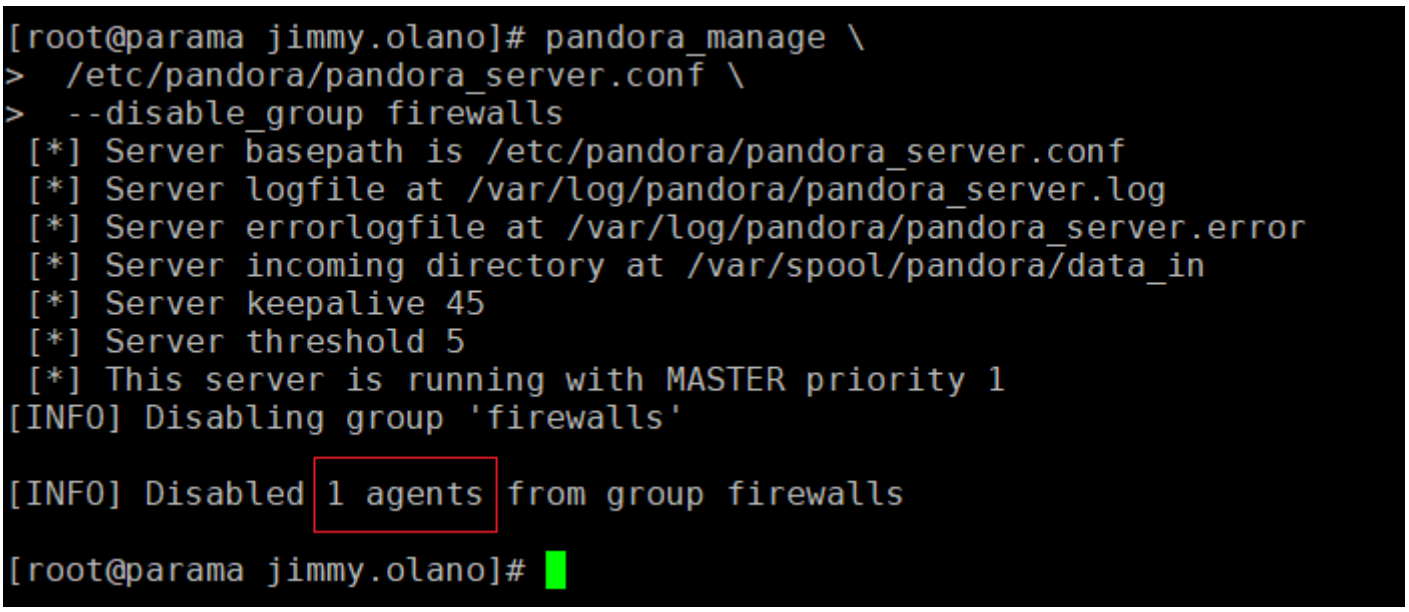
Description:

Disable all agents from an entire group. The agents of the group considered as parameter will be disabled with the execution of this option.

If you pass `All` as group, all agents from all groups will be disabled.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --disable_group Firewalls
```



enable_group

Required parameter:

- < group_name >

Description:

The agents of the group considered as parameter will be disabled with the execution of this option. If you pass `'All'` as group all agents from all groups will be enabled.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --enable_group firewalls
```

```
[root@parama jimmy.olano]# pandora_manage \
>  /etc/pandora/pandora_server.conf \
>  --enable_group FIREWALLS
 [*] Server basepath is /etc/pandora/pandora_server.conf
 [*] Server logfile at /var/log/pandora/pandora_server.log
 [*] Server errorlogfile at /var/log/pandora/pandora_server.error
 [*] Server incoming directory at /var/spool/pandora/data_in
 [*] Server keepalive 45
 [*] Server threshold 5
 [*] This server is running with MASTER priority 1
[INFO] Enabling group 'FIREWALLS'

[root@parama jimmy.olano]#
```

create_group

Required parameters:

- < group_name >

Optional paramaters:

- < parent_group_name >
- < icon >
    - applications
    - application_osx
    - application_osx_terminal
    - bricks
    - chart_organisation
    - clock
    - computer
    - database
    - database_gear
    - docker
    - drive_network
    - email
    - eye
    - firewall
    - heart
    - house
    - images
    - lightning
    - lock
    - network
    - plugin
    - printer
    - server_database
    - transmit
    - vmware

    - without_group
    - world
- < description >

Description:

A new group will be created if it doesn't exist and optionally, can be assigned a parent group, a icon (the icon name without extension, choose one from list above) and description. The parent group by default is `'All'` and the default icon is empty string (without icon).

If you need to add a description:

- If you will not use parent group set `All` as fill parameter.
- If no icon is to be used, place an empty string (two single quotes together) as the filler parameter.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --create_group \
    'New group name' \
    Web computer 'New description.'
```

delete_group

Required parameter:

- < group_name >

Description:

Delete an agent group.

Every agent belongs to indicated group will be deleted too.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf —-delete_group 'Group name'
```

update_group

Required parameters:

- < group_id >

Optional parameters:

- < group_name_to_modify >
- < parent_group_name >
- < icon >
    - applications
    - application_osx
    - application_osx_terminal
    - bricks
    - chart_organisation
    - clock
    - computer
    - database
    - database_gear
    - docker
    - drive_network
    - email
    - eye
    - firewall
    - heart
    - house
    - images
    - lightning
    - lock
    - network
    - plugin
    - printer
    - server_database
    - transmit
    - vmware
    - without_group
    - world
- < description >

Description: A group is modified by its numerical identifier. Parameters that can be modified:

- Name group.
- Name group parent
- Icon (only name, without extension, choose one from list above).
- Description.

- You can use the " get_agent_group_id " function to find the id of the group of any agent.
- At Web console go to menu Profiles→ Manage agent groups and take number id.

You must type exactly same existing parameters for only modify description field.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf –update_group 18 \
    'Group name' 'Web' 'transmit' 'Description'
```

stop_downtime

Required parameter:

- < downtime_name >

Description:

Stop a planned downtime. If the downtime is finished, a message will be showed. Returning messages:

- `[ERROR] Planned_downtime < downtime_name > cannot be stopped.`
- `[INFO] Planned_downtime < downtime_name > is already stopped.`
- `[INFO] Stopping planned downtime < downtime_name >`

Example (see call and syntax):

```
pandora_manage.pl /etc/pandora/pandora_server.conf --stop_downtime 'Downtime
name'
```

get_agent_status

Required parameter:

- < agent_name >

Optional parameter:

- `use_agent_alias`

  To identify the agent by its alias instead of its name,
  use `use_agent_alias` right at the end.

Description: Get the status of a given agent by its name or alias.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --get_agent_status 'agent name'
```

get_agent_group

Required parameter:

- < agent_name >

Optional parameter:

- `use_agent_alias`

  To identify the agent by its alias instead of its name,
  use `use_agent_alias` right at the end.

Description: Get the group of a given agent by its name or alias.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --get_agent_group 'agent_name'
```

get_agent_group_id

Required parameter:

- < agent_name >

Optional parameter:

- `use_agent_alias`

  To identify the agent by its alias instead of its name,
  use `use_agent_alias` right at the end.

Description: Get the identification group of a given agent by its name or alias.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --get_agent_group_id 'agent
name'
```

get_agent_modules

Required parameter:

- < agent_name >

Optional parameter:

- `use_agent_alias`

  To identify the agent by its alias instead of its name, use `use_agent_alias` right at the end.

Description: Get the modules (identifications and names, separated by a comma) of a given agent by its name or alias.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --get_agent_modules 'agent name'
```

get_agents

Required parameter: None. To get help for such a command you can use (see call and syntax):

```
 pandora_manage /etc/pandora/pandora_server.conf --get_agents '' '' '' '' '' ''
'' ''
```

You will get the following response indicating that it exceeds the number of parameters:

```
[ERROR] Parameters error: 8 received | 0-7 necessary.

Pandora FMS CLI 7.0NG.764 Build 220930 Copyright (c) 2013-2021 Artica ST
This program is Free Software, licensed under the terms of GPL License v2
You can download latest versions and documentation at http://www.pandorafms.org

[*] Pandora FMS module loaded.

Available options for --get_agents:

        --get_agents [<group_name> <os_name> <status> <max_modules>
<filter_substring> <policy_name> <use_alias>] : Get
          list of agents with optative filter parameters.
```

Optional parameters: Allow filtering the list of agents in a serialized format where the order matters, use two single quotes to skip parameter(s). They are numbered here for better control and visualization:

1. < group_name >
2. < os_name >
3. < status > Posible values: `critical`, `warning`, `unknown`, `normal`.
4. < number_max_of_modules >
5. < key_string > Allows filtering by agent name (or agent alias if you add at the end `use_agent_alias`).

6. < policy_name >
7. use_agent_alias

Description: Get all agents (without parameters) or agents according to a filter.

Examples (see call and syntax):

- Get all agents in the Network group:

```
pandora_manage /etc/pandora/pandora_server.conf --get_agents 'Network' '' '' ''
'' ''
```

- Get all the agents that have Linux as operating system:

```
pandora_manage /etc/pandora/pandora_server.conf --get_agents '' 'Linux' '' '' ''
''
```

- Obtain all agents whose status is normal:

```
pandora_manage /etc/pandora/pandora_server.conf --get_agents '' '' 'normal' ''
'' ''
```

- Obtain all agents with zero modules (no modules at all):

```
pandora_manage /etc/pandora/pandora_server.conf --get_agents '' '' '' '0' '' ''
```

- Get all agents with zero modules in the Servers group:

```
pandora_manage /etc/pandora/pandora_server.conf --get_agents 'Servers' '' '' '0'
'' ''
```

- Get all agents that contain the word test in their alias:

```
pandora_manage /etc/pandora/pandora_server.conf --get_agents '' '' '' '' 'test'
'' use_agent_alias
```

get_agents_id_name_by_alias

Parameters: <agent_alias> [strict]

Description: List id and alias of agents matching given alias. Using strict limits search only to exact alias.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
get_agents_id_name_by_alias 'station'
```

Delete_conf_file

(>=5.0)

Parameters: <agent_name> <use_agent_alias>

Description: The conf file of one agent will be deleted

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --delete_conf_file
'Agent name'
```

Clean_conf_file

(>=5.0)

Parameters: [<agent_name> <use_agent_alias>]

Description: The conf file of one or all agents (without parameters) will be cleaned (All modules, policies, file collections and comments will be deleted).

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --clean_conf_file 'Agent
name'
```

get_bad_conf_files

Parameters: None.

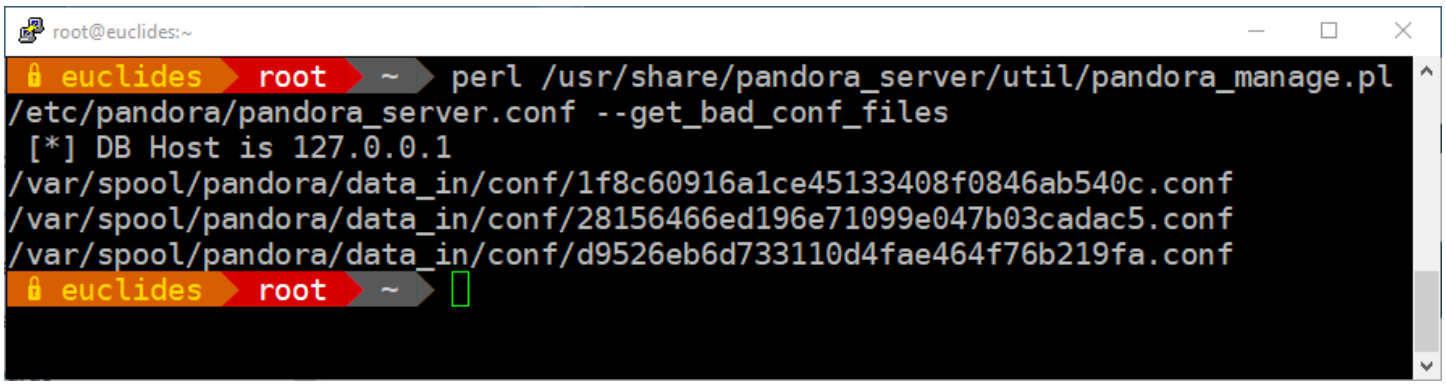
Description:

1. The misconfigured configuration files will be listed by searching the main *tokens*:
    - `server_ip`
    - `server_path`
    - `temporary`
    - `logfile`
2. In case there are corrupt files that cannot be opened: `[WARN] Can't open file < path_file >`.
3. In case of not finding the file in the path: `[WARN] File not exist < path_file >`.
4. In case the configuration files are correct (according to the *tokens* of the first point): `[INFO] No bad files found`

Examples:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_bad_conf_files
```

Example output:



migration_agent_queue

(>=7.21) only for metaconsole

Parameters: [id of the agent to migrate][name of the source node][name of the destination node][optional parameter by default this 0 will migrate the data of both the history and the db itself if it is at 1 it will only migrate those of the db itself]

Description: Add an agent to the migration queue to move from node to another node this is for the metaconsole only. To do it correctly, you will need to fill in the API configuration parameters in pandora_server.conf.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --migration_agent_queue
1 nodo1 nodo2 0
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

migration_agent

(>=7.21) only for metaconsole Parametrizes: [id of the agent to check]

Description: it will return true or false depending on whether the agent entered exists in the agent migration table

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --migration_agent 1
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

create_downtime

Parameters: <downtime_name> <description> <date_from> <date_to> <id_group> <monday (0|1)> <tuesday (0|1)><wednesday (0|1)> <thursday (0|1)> <friday (0|1)> <saturday (0|1)> <sunday (0|1)> <periodically_time_from HH:MM:SS><periodically_time_to HH:MM:SS> <periodically_day_from> <periodically_day_to><type_downtime> <type_execution> <type_periodicity> <id_user>

Description: Created Planned downtime.

- You must have configured data API in `pandora_server.conf`.
- The date format must be `MM/DD/YYYY` (Month/Day/Year) for this call to work properly.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_downtime
Testing Testing 05/07/2022 06/07/2022 0 1 1 1 1 1 1 1 17:07:00 18:08:00 1 31
quiet periodically weekly 7
```

add_item_downtime

Required parameters:

- < id_parada_planificada >
- < id_agente1,id_agente2,id_agente3,...,id_agenteN >
- < nombre_módulo1,nombre_módulo2,nombre_módulo3,...,nombre_móduloN >

Description:

Records will be added to a scheduled stop with the submitted data. Write the list of agent identifiers and the list of module names without leaving spaces between the commas that separate the components of both lists. If a module name contains spaces, enclose the module name in two single quotes. If two or more agents are passed, it is assumed by default that you want to add common modules, otherwise it will display a warning message.

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --add_item_downtime 1 1
Status,'Available Memory'
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

get_all_planned_downtimes

Required parameters:

- < name_to_search >

Optional parameters:

- < id_group >
- < type_downtime >
  - `quiet`
  - `disable_agents`
  - `disable_agents_alert`
- < type_execution >
  - `once`
  - `periodically`
- < type_periodicity >
  - `weekly`
  - `monthly`

Description:

List all matches of planned downtime. See Scheduled downtimes options.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --get_all_planned_downtimes ''
```

get_planned_downtimes_items

Required parameters:

- < name_to_search >

Optional parameters:

- < id_group >
- < type_downtime >
  - quiet
  - disable_agents
  - disable_agents_alert
- < type_execution >
  - once
  - periodically
- < type_periodicity >
  - weekly
  - monthly

Description:

List all of items that matches planned downtime name. See Scheduled downtimes options.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --get_planned_downtimes_items
'Name downtime'
```

set_planned_downtimes_deleted

Required parameter:

- < id_planned_downtime >

Description:

Delete a planned downtime especified by ID. Some return messages:

```
This planned downtime is deleted

Problems with this planned downtime

The scheduled downtime is still being executed
```

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --set_planned_downtimes_deleted
```

```
7
```

set_disabled_and_standby

Only for Command Center (Metaconsole) and Nodes.

Required parameters:

- < id_agent >
- *For Command Center (Metaconsole)* < id_metaconsole >, *for nodes* put any value, is indifferent.

Optional parameter:

- < value> By default 1 for enable agent, 0 for disabled.

Description: Disable an agent and, if it has remote configuration, set to standby mode.

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

Example (see call and syntax):

- Example to enable an agent from the node:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
set_disabled_and_standby 2 0 1
```

- Example to disable an agent from the node:

```
pandora_manage /etc/pandora/pandora_server.conf --set_disabled_and_standby 2 0 0
```

- Example to enable an agent from the metaconsole:

```
pandora_manage /etc/pandora/pandora_server.conf --set_disabled_and_standby 2 1 1
```

- Example to disable an agent from the metaconsole:

```
pandora_manage /etc/pandora/pandora_server.conf --set_disabled_and_standby 2 1 0
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

reset_agent_counts

Parameters: <id_agent>

Description: Synchronizes module counts and alerts for a given agent. If you want to synchronize with all agents, the first parameter must be “All”.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --reset_agent_counts
"All"
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

insert_gis_data

Required parameters:

- < id_of_agent >
- < latitude >
- < longitude >
- < altitude >

Optional parameters:

None.

Description: Updates an agent's GIS data.

Examples (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --insert_gis_data 1 2 3 4
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

get_gis_agent

Required parameters:

- < agent_id >

Optional parameters:

- None.

Description: Gets an agent's GIS data.

Examples (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --get_gis_agent 1
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

agent_set_os

Required parameters:

- < agent_id >
- < operating_system_id >
- < operating_system_version >

Optional parameters:

- None.

Description: Add or change operating system (and obsolescence) to an agent.

Examples (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --agent_set_os 7 3 'Ubuntu 16.04'
```

Modules

create_data_module

Required parameters:

- < module_name >
- < module_type >
- < agent_name >
- < description > To omit this parameter, use “” .
- < module_group > To omit this parameter, use “” .
- < min > To omit this parameter, use “” .
- < max > To omit this parameter, use “” .

Optional parameters:

1. < post_process >
2. < interval >
3. < warning_min >
4. < warning_max >
5. < critical_min >
6. < critical_max >
7. < history_data >
8. < definition_file >
9. < warning_string >
10. < critical_string >
11. < enable_unknown_events >
12. < ff_threshold >
13. < each_ff >
14. < ff_threshold_normal >
15. < ff_threshold_warning >
16. < ff_threshold_critical >
17. < ff_timeout >
18. < warning_inverse >
19. < critical_inverse >
20. < critical_instructions >
21. < warning_instructions >
22. < unknown_instructions >
23. `use_agent_alias`: For the agent search use the agent alias instead of the agent name.
24. `ignore_unknown`: Discards the unknown state calculation, so that the module will never change to that state and will always keep the last known state.
25. < number_of_intervals_in_warning > Allows state scaling by specifying the maximum number of consecutive intervals in which the module remains in warning state. If this value is exceeded, the module will escalate to critical status.

Description: A module kind data will be created in an agent with the module name, kind of module and name of the agent (see option `use_agent_alias`) where it will be created. Optionally it will be possible to give a description, the module group, etc.

The default values are `0` for the minimum and maximum values, history_data and post_process; `300` for interval.

Example (see call and syntax):

Simple example for state scaling:

```
pandora_manage /etc/pandora/pandora_server.conf --create_data_module
data_module_test generic_data
3f9c41953a072afa229aa0a7fe3a6203a1ecf86b40b8a13d8a7d9916f0210bb1 "" "" "" "" ""
"" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" 2001
```

Complex example, the module definition file will contain something like this:

```
module_begin
module_name My module
module_type generic_data
module_exec cat /proc/meminfo  | grep MemFree | awk '{ print $2 }'
module_end
```

```
pandora_manage /etc/pandora/pandora_server.conf --create_data_module 'My module'
generic_data 'My agent' 'Module description' 'General' 1 3 0 300 0 0 0 0 1
/home/user/definition_file 'warning text' 'critical text'
```

If you introduce a different name or kind between the parameters and the file definition, the fixed on the file will have priority.

create_web_module

Required parameters:

- < module_name >
- < module_kind >
- < agent_name >

Optional parameters:

1. < description >
2. < module_group >
3. < min >
4. < max >
5. < post_process >
6. < interval >
7. < warning_min >
8. < warning_max >
9. < critical_min >
10. < critical_max >
11. < history_data >
12. < retries >
13. < requests >
14. < agent_browser_id >
15. < auth_server >
16. < auth_realm >

17. < definition_file >
18. < proxy_url >
19. < proxy_auth_login >
20. < proxy_auth_password >
21. < warning_str >
22. < critical_str >
23. < enable_unknown_events >
24. < ff_threshold >
25. < each_ff >
26. < ff_threshold_normal >
27. < ff_threshold_warning >
28. < ff_threshold_critical >
29. < ff_timeout >
30. < warning_inverse >
31. < critical_inverse >
32. < critical_instructions >
33. < warning_instructions >
34. < unknown_instructions >
35. < `use_agent_alias` >: For the agent search use the agent alias instead of the agent name.
36. < `ignore_unknown` >: Discards the unknown state calculation, so that the module will never change to that state and will always keep the last known state.
37. < number_of_intervals_in_warning > Allows state scaling by specifying the maximum number of consecutive intervals in which the module remains in warning state. If this value is exceeded, the module will escalate to critical status.

Description: A web data module will be created in an agent with the module name, kind of module and name of the agent where it will be created (see parameter `use_agent_alias`) . Optionally it will be possible to give a description, the module group, min and max values, a post_process value, an interval in seconds, min and max warning values, min and max critical values, a history data value and one module definition file among others.

Example (see call and syntax):

Simple example for state scaling:

```
pandora_manage /etc/pandora/pandora_server.conf --create_web_module
web_module_test web_data
3f9c41953a072afa229aa0a7fe3a6203a1ecf86b40b8a13d8a7d9916f0210bb1 "" "" "" "" ""
"" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
"" "" "" "" 2001
```

As a complex example, the module definition file will contain something like this:

```
task_begin
get http://pandorafms.com
task_end
```

```
pandora_manage /etc/pandora/pandora_server.conf --create_web_module
'module_name' web_data 'pandorafms' 'Module description' 'General' 0 100 0 300 0
0 0 0 1 0 1 'Pandora FMS' auto public /home/user/file_definition
```

```
'http://proxy.url' 'proxy_login' 'proxy_password' 1 10 10 10 10 10 10 10 10 1 1
'critical_instructions' 'warning_instructions' 'unknown_instructions'
```

create_network_module

Required parameters:

- < module_name >
- < module_type >
- < agent_name >
- < module_address >

Optional parameters:

- < module_port > numerical value between 1 and 65535.
- < description >
- < module_group >
- < min >
- < max >
- < post_process >
- < interval >
- < warning_min >
- < warning_max >
- < critical_min >
- < critical_max >
- < history_data >
- < ff_threshold >
- < warning_str >
- < critical_str >
- < enable_unknown_events >
- < each_ff >
- < ff_threshold_normal >
- < ff_treshold_warning >
- < ff_threshold_critical >
- < timeout >
- < retries >
- < critical_instructions >
- < warning_instructions >
- < unknown_instructions >
- < warning_inverse >
- < critical_inverse >
- `use_agent_alias` to search for the agent by its alias instead of its name.
- `ignore_unknown` Discards the unknown state calculation, so that the module will never change to that state and will always keep the last known state.
- < number_of_intervals_in_warning > Allows state scaling by specifying the maximum number of consecutive intervals in which the module remains in warning state. If this value is exceeded, the module will escalate to critical status.

Description: A network module will be created in an agent with the module name, module type, agent name where it will be created and the module address specified. Optionally, it can be given

a port number, a description, minimum and maximum values, a post processing ' value, an interval in seconds, minimum and maximum warning values, minimum and maximum criticality values, and a historical data value.

The default values are `0` for minimum and maximum, `history_data` and `post_process` and `300` seconds for the interval.

The port number is optional, since ICMP type modules do not require it. For all other types, a module must be specified.

Example (see call and syntax):

Simple example for state scaling:

```
pandora_manage /etc/pandora/pandora_server.conf --create_network_module
net_module_test remote_tcp
3f9c41953a072afa229aa0a7fe3a6203a1ecf86b40b8a13d8a7d9916f0210bb1 127.0.0.1 "1"
"" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
"" 2001
```

Another example:

```
pandora_manage /etc/pandora/pandora_server.conf --create_network_module 'My
module' remote_tcp 'My agent' 192.168.12.123 8080 'Module description' 'General'
1 3 0 300 0 0 0 0 1
```

Create_network_component

Parameters: <network_component_name> <network_component_group> <network_component_type> [<description> <module_interval> <max_value> <min_value> <snmp_community> <id_module_group> <max_timeout> <history_data> <min_warning> <max_warning> <str_warning> <min_critical> <max_critical> <str_critical> <min_ff_event> <post_process> <disabled_types_event> <each_ff> <min_ff_event_normal> <min_ff_event_warning> <min_ff_event_critical>]

Description:

A network component will be created. Optionaly, it will be possible to give it a port, a description, values min and max, a post_process value, an interval in seconds, a warning min and max values, critical min and max values and a history data value.

The default values are 0 for the min and max, history_data and post_process an another 300 for the interval.

the port is optional, so the modules kind ICMP don't need it. In the rest of kinds, it is necessary to

specify one module.

Notes:

The next parameters are only for the Pandora version 5 and next versions:

- <enable_unknown_events>

The next parameters are only for the Pandora version 5.1 and next versions:

- <each_ff>
- <ff_threshold_normal>
- <ff_threshold_warning>
- <ff_threshold_critical>

Example:

```
perl /tmp/pandorafms/pandora_server/util/pandora_manage.pl
/etc/pandora/pandora_server.conf --create_network_component "example_name" 2 7
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

create_snmp_module

Required parameters:

- < module_name >
- < module_type >
- < agent_name >
- < module_address >
- < module_port >
- < version >

Optional parameters:

- < community >
- < oid >
- < description >
- < module_group >
- < min >
- < max >
- < post_process >
- < interval >
- < warning_min >
- < warning_max >

- < critical_min >
- < critical_max >
- < history_data >
- < snmp3_priv_method >
- < snmp3_priv_pass >
- < snmp3_sec_level >
- < snmp3_auth_method >
- < snmp3_auth_user >
- < snmp3_auth_pass >
- < ff_threshold>
- < warning_str >
- < critical_str >
- < unknown_events >
- < each_ff >
- < ff_threshold_normal >
- < ff_threshold_warning >
- < ff_threshold_critical >
- < timeout >
- < retries >
- `use_alias` to search for the agent by its alias instead of its name.
- `ignore_unknown` Discards the unknown state calculation, so that the module will never change to that state and will always keep the last known state.
- < critical_instructions >
- < warning_instructions >
- < unknown_instructions >
- < warning_inverse >
- < critical_inverse >
- < number_of_intervals_in_warning > Allows state scaling by specifying the maximum number of consecutive intervals in which the module remains in warning state. If this value is exceeded, the module will escalate to critical status.

Description: An SNMP type module will be created in an agent with the module name, module type, name of the agent where it will be created, the module address, the associated port and the SNMP version specified. Optionally, a community name, OID, description, module group, minimum and maximum values, post processing value, interval in seconds, minimum and maximum warning values, minimum and maximum criticality values, and SNMP 3 parameters such as methods, passwords, etc. can be provided.

The default values are `0` for minimum and maximum, `history_data` and `post_process` and 300 seconds for the interval.

Example (see call and syntax):

Simple example for state scaling:

```
pandora_manage /etc/pandora/pandora_server.conf --create_snmp_module
snmp_module_test remote_snmp_inc
3f9c41953a072afa229aa0a7fe3a6203a1ecf86b40b8a13d8a7d9916f0210bb1 127.0.0.1 "1"
"3" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
"" "" "" "" "" "" "" "" "" "" 2001
```

Another example:

```
pandora_manage /etc/pandora/pandora_server.conf --create_snmp_module 'My module'
remote_snmp_inc 'My agent' 192.168.12.123 8080 1 my_comunnity my_oid 'Module
description'
```

create_plugin_module

Required parameters:

- < module_name >
- < module_type >
- < agent_name >
- < module_address >
- < module_port > numerical value between 1 and 65535.
- < plugin_name >
- < user >
- < password >
- < parameters >

Parámetros opcionales:

- < description >
- < module_group >
- < min >
- < max >
- < post_process >
- < interval >
- < warning_min >
- < warning_max >
- < critical_min >
- < critical_max >
- < history_data >
- < ff_threshold >
- < warning_str >
- < critical_str >
- < enable_unknown_events >
- < each_ff >
- < ff_threshold_normal >
- < ff_treshold_warning >
- < ff_threshold_critical >
- < timeout >
- < critical_instructions >
- < warning_instructions >
- < unknown_instructions >
- < warning_inverse >
- < critical_inverse >
- `use_agent_alias` to search for the agent by its alias instead of its name.
- `ignore_unknown` Discards the unknown state calculation, so that the module will never change to that state and will always keep the last known state.

- < number_of_intervals_in_warning > Allows state scaling by specifying the maximum number of consecutive intervals in which the module remains in warning state. If this value is exceeded, the module will escalate to critical status.

Description: A module of type *plugin* will be created in an agent with the module name, module type, name of the agent where it will be created, the module address, the associated port and the corresponding plugin name. Optionally, a description, the module group, minimum and maximum values, a post processing value, an interval in seconds, minimum and maximum warning values, minimum and maximum criticality values, among other optional values, can be provided.

The default values are `0` for minimum and maximum, `history_data` and `post_process` and `300` for the interval.

Example (see call and syntax):

Simple example for state scaling:

```
pandora_manage /etc/pandora/pandora_server.conf --create_plugin_module
plugin_module_test generic_data
3f9c41953a072afa229aa0a7fe3a6203a1ecf86b40b8a13d8a7d9916f0210bb1 127.0.0.1 "1"
"DNS Plugin" "admin" "pandora" "100" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
"" "" "" "" "" "" "" "" "" "" "" "" "" 2001
```

Another example:

```
pandora_manage /etc/pandora/pandora_server.conf --create_plugin_module 'My
module' generic_data 'Mi agente' 192.168.12.123 8080 myplugin myuser mypass
'param1 param2 param3' 'Descripcion del modulo' 'General' 1 3 0 300 0 0 0 0 1
```

Get_module_group

Parameters: [<nombre del grupo de módulo>]

Description: Displays the available module groups. We can filter by adding the name of the group or part of it.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_module_group
"grupo de módulos"
```

Create_module_group

Parameters: <module_group_name>

Description: A module group will be created with the module_group_name

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_module_group
'My module group'
```

Module_group_synch

Parameters: <server_name_1|server_name_2|...server_name_n> [<return_type>]

Description: The groups of modules of the goal will be synchronized with the nodes that have been specified in the first parameter, separated with “|”. Optionally you can indicate in the second parameter the format in which the data will be returned (csv or json). By default they will come in csv format.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --module_group_synch
"server_name1|server_name2|server_name3" "json"
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

Create_synthetic

Parameters: <module_name> <synthetic_type> <agent_name> <opts> <use_agent_alias>

Where <synthetic_type> can be “arithmetic” or “average”.

Examples

```
pandora_manage /etc/pandora/pandora_server.conf --create_synthetic
example_module arithmetic example_agent <opts>
pandora_manage /etc/pandora/pandora_server.conf --create_synthetic
example_module average example_agent <opts>
```

Let's explain all the posible parameters that can be used on the <opts»

- <opts> = <source_agent1>,<operator>,<source_module1> <source_agent2>,<operator>,<source_module2>
- <opts> = <source_agent1>,<operator>,<source_module1> <operator>,<fixed_value>

Examples

```
pandora_manage /etc/pandora/pandora_server.conf --create_synthetic
example_module arithmetic example_agent my_agent,+,my_module my_agent2,-
,my_module2
pandora_manage /etc/pandora/pandora_server.conf --create_synthetic
example_module arithmetic example_agent my_agent,+,my_module *,10
pandora_manage /etc/pandora/pandora_server.conf --create_synthetic
example_module average example_agent my_agent,x,my_module my_agent,x,my_module2
```

On the first example, we will create in the agent *example_agent* an arithmetic type of module named *example_module*. Its content will be the result of the operation: *my_module - my_module2*. On the second example, we will create in the agent *example_agent* an arithmetic type of module named *example_module*. Its content will be the result of the operation: *my_module * 10*. On the third example, we will create in the agent *example_agent* an average type of module named *example_module*. Its content will be the average value betwee *my_module* and *my_module2*.

The first operator that is used, on the previous case between *my_agent* and *my_module* (+), is ignored and doesn't affect to the result of the operations, but it is needed due to the internal structure of the tool. It's very important to keep into account that the arithmetic operation will be done with the next (in this case, the second) operator, on the current example, between *my_agent2* and *my_module2*, or right before the value *10* (in this case, operators “-” or “*”). The possible operators are:

- + sum
- - substraction
- * multiplication
- / division
- x average. This operator only can be used with the *average* type.

Note:

Available in Pandora 5.1 SP4 and above

Example:

pandora_manage /etc/pandora/pandora_server.conf –create_synthetic memory_sum arithmetic nodo51sp3 production1,+,AvailableMemory production2,+,AvailableMemory

```
pandora_manage /etc/pandora/pandora_server.conf --create_synthetic
traffic_difference arithmetic nodo51sp3 nodo51sp3,+,InputTraffic nodo51sp3,-
,OutputTraffic
```

delete_module

Required parameters:

- < module_name >
- < agent_name >

Optional parameter:

- use_alias

Description: A module will be removed from an agent by passing the name of both as a parameter. In case this module is in a local agent, it will also be deleted from the configuration file. From version 771 on wards, the *descendants* of the deleted module are deleted recursively.

Examples (see call and syntax):

```
pandora_manage.pl /etc/pandora/pandora_server.conf --delete_module 'My module'
'My agent'
```

Data_module

Parameters: <server_name> <agent_name> <module_name> <module_type> <module_new_data> [<datehour>] <use_agent_alias>

Description: It'll be send data to an agent module giving it as parameter the server name, the agent, the module name, the type of module amd the new data to be inserted. Optionally, it'll be possible to send the date-hour that will be as that of the data sending with 24 hours format: 'YYY-MM-DD HH:mm'. In the case of not sending this parameter, the current data will be shown.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --data_module Servidor-
General 'My agent' 'My modulo' 'generic_data' 1 '2010-05-31 15:53'
```

Get_module_data

(>=5.0)

Parameters: <agent_name> <module_name> <interval> [<csv_separator>] <use_agent_alias>

Description: Will be returned the data of a module as 'timestamp data' in CSV format of the last X seconds (interval) using as default separator ';'

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_module_data 'agent
name' 'module name' 86400 ':'
```

Get_module_id

Parameters: <agent_id> <module_name>

Description: The id of a module of an agent is returned.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_module_id 4 'host
alive'
```

Get_module_custom_id

Parameters: <agentmodule_id>

Description: The value of the field *Custom ID* of a specific module of an agent will be returned.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_module_custom_id 4
```

Set_module_custom_id

Parameters: <agentmodule_id> [<custom_id>]

Description: The value of the field *Custom ID* of a specific module of an agent will be inserted. If it is left blank it will eliminate the content that would have previously left the field empty.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --set_module_custom_id 4
'21'
```

Delete_data

Parameters: <module_name> <agent_name> | -a <agent_name> | -g <group_name> <use_agent_alias>

Description: All data associated to a module will be deleted from the historical data in case of having as parameter -m and the name of this one and its agent name; from the agent modules if as parameter the option '-a' is given, and the agent or modules name of all agents from a group, if as parameter the option '-g' and the group name is given.

Example:

perl pandora_manage.pl /etc/pandora/pandora_server.conf –delete_data -a 'My agent'

In this example all historic data will be deleted from all modules that belongs to the 'My agent' agent.

Update_module

Parameters: <module_name> <agent_name> <field_to_update> <new_value> <use_agent_alias>

Description: A given field of an existent data module will be updated. The module type will be detected to allow update the specific fields for each type.

The possible fields are the following:

- Common to any module: module_name, agent_name, description, module_group, min, max, post_process, history_data, interval, warning_min, warning_max, critical_min, critical_max, warning_str, critical_str, ff_threshold, each_ff, ff_threshold_normal, ff_threshold_warning, ff_threshold_critical, critical_instructions, warning_instructions, unknown_instructions
- For the data modules: ff_timeout
- For the network modules: module_address, module_port
- For the SNMP modules: module_address, module_port, version, community, oid, snmp3_priv_method, snmp3_priv_pass, snmp3_sec_level, snmp3_auth_method, snmp3_auth_user, snmp3_priv_pass
- For the plugin modules: module_address, module_port, plugin_name, user, password, parameters

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --update_module 'Module name' 'Agent name' description 'New description'
```

add_tag_to_module

Parameters: <agent_name> <module_name> <tag_name>

Description: Add a tag to the given module.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --add_tag_to_module 'Agent name' 'Module name' 'Tag name'
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements)

: `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

Get_agents_module_current_data

(>=5.0)

Parameters: <module_name>

Description: Get the agent and current data of all the modules with a given name.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
get_agents_module_current_data 'Module name'
```

Create_network_module_from_component

(>=5.0)

Parameters: <agent_name> <component_name> <use_agent_alias>

Description: Create a new network module in the specified agent from a network component.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
create_network_module_from_component 'Agent name'
```

Create_data_module_from_local_component

(>=5.1)

Parameters: <agent_name> <component_name> <use_agent_alias>

Description: Create a new data module in the specified agent from a local component.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
create_data_module_from_local_component 'Agent name' 'My local component'
```

Create_local_component

(>=5.1)

Parameters: <component_name> [<description> <id_os> <os_version> <id_network_component_group> <type> <min> <max> <module_interval> <id_module_group> <history_data> <min_warning> <max_warning> <str_warning> <min_critical> <max_critical> <str_critical> <min_ff_event> <post_process> <unit> <wizard_level> <critical_instructions> <warning_instructions> <unknown_instructions> <critical_inverse> <warning_inverse> <id_category> <disabled_types_event> <tags> <min_ff_event_normal> <min_ff_event_warning> <min_ff_event_critical> <each_ff> <ff_timeout>]

Description: Create a new local component.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_local_component
'New local component' 'module_begin\nmodule_name name\nmodule_type
generic_data\nmodule_exec exec\nmodule_end'
```

Apply_module_template

Parameters: [<id_template> <id_agent>]

Description: Tis module templates are an agrupation that contenais network modules. This templates can be applied directly to agents, without having to include each module onw by one.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --apply_module_template
1 1
```

Alerts

Create_template_module

Parameters: <template_name> <module_name> <agent_name> <use_agent_alias>

Description: A template will be assigned to an agent module giving it the template name, the module and the agent as parameters.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_template_module
template001 'My module' 'My agent'
```

Delete_template_module

Parameters: <template_name> <module_name> <agent_name> <use_agent_alias>

Description: it'll be unassigned a module template of one agent giving it the template name, the module and the agent as parameters.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --delete_template_module
plantilla001 'Mi modulo' 'Mi agente'
```

Create_template_action

Parameters: <action_name> <template_name> <module_name> <agent_name> [<fires_min> <fires_max>] <use_agent_alias>

Description: It'll be added an action to an alert giving as parameter the name of the action and that of the template, module and agent that composes the alert. It'll be also possible giving it in an optional way the values of scaling fires_min and fires_max (by default 0).

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_template_action
action012 template001 'My module' 'My agent' 0 4
```

Delete_template_action

Parameters: <action_name> <template_name> <module_name> <agent_name> <use_agent_alias>

Description: It'll be added an action to an alert giving as parameters the names of the action, template, module and agent that composes the alert.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --delete_template_action
action012 template001 'My module' 'My agent'
```

Disable_alerts

Parameters: No

Description: All alerts will be disabled with the execution of this option.If when it's executed we have any alert disabled and we activate all again, this one will be also enabled.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --disable_alerts
```

Enable_alerts

Parameters: No

Description: All the alerts will be activated with the execution of this option. If when it's executed we had any alert enabled and we disabled all again, this one will be also disabled.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --enable_alerts
```

Create_alert_template

Parameters: <template_name> <condition_type_serialized> <time_from> <time_to> [<description> <group_name> <field1> <field2> <field3> <priority> <default_action> <days> <time_threshold> <min_alerts> <max_alerts> <alert_recovery> <field2_recovery> <field3_recovery> <condition_type_separator>]

Description: An alert template will be created.

The field <condition_type_serialized> is the type options of the template serialized with the separator ';' by default. It's possible change the separator with the parameter <condition_type_separator> to avoid conflicts some options if it could contain the default character.

The possibilities are the following:

NOTE: In this examples is used the default separator ';' and the field matches_value is a binary value to set if the alert will be fired when the value match or when the value not match with the conditions.

- Regular expression:
  - Syntaxis: <type>;<matches_value>;<value>

    - Example: regex;1;stopped|error (Alert when value match regexp 'stopped|error')
- Max and min:
    - Syntaxis: <type>;<matches_value>;<min_value>;<max_value>
    - Example: max_min;0;30;50 (Alert when value is out of interval 30-50)
- Max.:
    - Syntaxis: <type>;<max_value>
    - Example: max;70 (Alert when value is above 70)
- Min.:
    - Syntaxis: <type>;<min_value>
    - Example: min;30 (Alert when value is below 30)
- Equal to:
    - Syntaxis: <type>;<value>
    - Example: equal;0 (Alert when value is equal to 0)
- Not equal to:
    - Syntaxis: <type>;<value>
    - Example: not_equal;100 (Alert when value is not equal to 100)
- Warning status:
    - Syntaxis: <type>
    - Example: warning (Alert when status turns into warning)
- Critical status:
    - Syntaxis: <type>
    - Example: critical (Alert when status turns into critical)
- Not normal:
    - Syntaxis: <type>
    - Example: not_normal (Alert when status differs from normal)
- Unknown status:
    - Syntaxis: <type>
    - Example: unknown (Alert when status turns into unknown)
- On Change:
    - Syntaxis: <type>;<matches_value>
    - Example: onchange;1 (Alert when value changes)
- Always:
    - Syntaxis: <type>
    - Example: always (Alert all times)

The field <days> is seven binary characters that specify the days of the week when the alert will be activated. i.e.: 0000011 to activate the alert only Saturday and Sunday.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf.2011-10-25 --
create_alert_template 'template name' "max_min@1@3@5" 09:00 18:00 "Email will be
sended when the value is in the interval 3-5, between 9AM and 6PM, and only the
Mondays. Separator is forced to @" "Unknown" "mail@mail.com" "subject" "message"
3 "Mail to XXX" 1000000 38600 1 2 0 // // @
```

Delete_alert_template

(>=5.0)

Parameters: <template_name>

Description: An alert template will be deleted if exists.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --delete_alert_template
'Template name'
```

Update_alert_template

(>=5.0)

Parameters: <template_name> <field_to_update> <new_value>

Description: A given field of an existent alert template will be updated. The possible fields are the following: name, description, type, matches_value, value, min_value, max_value, time_threshold(0-1), time_from, time_to, monday(0-1), tuesday(0-1), wednesday(0-1), thursday(0-1), friday(0-1), saturday(0-1), sunday(0-1), min_alerts, max_alerts, recovery_notify(0-1), field1, field2, field3, recovery_field2, recovery_field3, priority(0-4), default_action, group_name.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --update_alert_template
'Template name' priority 4
```

Get_alert_actions

(>=5.0)

Parameters: [<action_name> <separator> <return_type>]

Description:Returns all action alerts. Optionally you can filter the result by name. Optionally in the second parameter you can specify the separator of the results and in the third parameter the format of the returned list (csv, json, string).

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_alert_actions
'Action 1' ';' 'json'
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements)

: `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

Get_alert_actions_meta

(>=5.0)

This feature is in Command Center (Metaconsole).

Parameters: [<server_name> <action_name> <separator> <return_type>]

Description: returns all alert actions of the meta console. Optionally you can filter the result by the name of the node or by the name of the action. Optionally in the second parameter you can specify the results separator and in the third parameter the format of the returned list (csv, json, string).

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_alert_actions_meta
// 'Action 1' ';' 'json'
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

Create_alert_command

(>=5.0)

Parameters: <name> <command> [<group|name> <description> <internal> <field_description> <field_value>]

Description: A command will be created with the specified name and command data. Optionally you can specify:

- Group name. If no group is provided, it will be assigned in the group 'All'.
- Command Description.
- Internal (1-0).
- Field description. This will be formatted as follows: '[“descr1”,“descr2”,“descr3”,“descr4”,“descr5”]'.
- Field value: This will be formatted as follows: [“val1”,“val2”,“val3”,“val4”,“val5”].

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf  --create_alert_command
'PRUEBA' 'command' 'All' 'Desc' '1'
'["des1","des2","des3","","des5","","","","",""]'
'["val1","val2","val3","val4","","","","","",""]'
```

Get_alert_commands

(>=5.0)

Parameters: [<name|<command> <group name> <description> <internal>]

Description: Allows you to view all alerts. Optionally you can filter the result by: name, command, group, description or if it is internal (1-0). In the case of the name, command and description it is not necessary to enter the exact data.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf  --get_alert_commands
"command_name"
```

validate_alert

Parameters: <template_name> <agent_id> <module_id> [`use_alias`]

Description: Validate an alert given a template name, an agent ID and a module ID. Instead use only an agent ID you may use name agent with last parameter `use_alias` (maybe for validate multiple alerts).

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf  --validate_alert 7 10
15
```

Validate_all_alerts

(>=5.0)

Parameters: None

Description: Validate all the alerts.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --validate_all_alerts
```

Create_special_day

(>=5.1)

Parameters: <special_day> <same_day> <description> <group_name>

Description: Create a special day. The possible same_days are monday, tuesday, wednesday, thursday, friday, saturday and sunday.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_special_day
2014-05-03 sunday Desc All
```

Delete_special_day

(>=5.1)

Parameters: <special_day>

Description: Delete specified special day.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --delete_special_day
2014-05-03
```

set_event_storm_protection

Parameters: <value>

Description: Enable (1) or disable (0) event storm protection.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
set_event_storm_protection 1
```

Update_special_day

(>=5.1)

Parameters: <special_day> <field_to_change> <new_value>

Description: Update specific field of a special day with new value. The possible fields are same_day, description and group_name. When same_day is set, possible new_values are monday, tuesday, wednesday, thursday, friday, saturday and sunday.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --update_special_day
2014-05-03 same_day monday
```

Users

create_user

NG 766 version or later: This function also works in
Command Center (Metaconsole).

Required parameters:

- < id_user >
- < password >
- < is_admin >

Optional parameters:

- < comments >

Description:

A user will be created with the user ID (which will also be the user name) and the password received. In addition a binary value (< is_admin > 0 false or 1 true) will be received specifying whether the user will be an superadmin or not. Optionally a comment about the created user can be added.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --create_user user9500 'user
9500' 0 'This user has the password "user 9500"'
```

delete_user

NG 766 or later: This function also works in Command Center (Metaconsole).

Required parameter:

- <id_user >

Description:

An user will be eliminated giving its name as parameter.

Example (see call and syntax):

```
pandora_manage.pl /etc/pandora/pandora_server.conf --delete_user user02
```

update_user

NG 766 or later: This function also works in Command Center (Metaconsole).

Required parameters:

- < id_user >
- < field_to_update >
- < new_value >

Description:

A given field of an existent user will be updated. The possible fields are the following:

- `email`.
- `phone`.
- `is_admin` ¿Is superadmin? (`0` false,`1` true).
- `language` , possible values:
  - `default`, English US.
  - `ca`, Catalan.
  - `en_GB`, English UK.
  - `es`, Spanish.
  - `fr`, French.
  - `ru`, Russian.
  - `ja`, Japanese.
  - `zh_CN` , Chinese.
- `comments`.
- `fullname`.

- password.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --update_user 'user 9500'
password 'XYZ'
```

enable_user

NG 766 or later: This function also works in Command Center (Metaconsole).

Required parameter:

- < user_id >

Description:

An existent user will be enabled. If it's already enabled, will showed only a message.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --enable_user 'user 9500'
```

disable_user

NG 766 or later: This function also works in Command Center (Metaconsole).

Required parameter:

- < user_id >

Description:

An existent user will be disabled. If it's already disabled, will showed only a message.

Example (see call and syntax):

```
pandora_manage.pl /etc/pandora/pandora_server.conf --disable_user 'user 9500'
```

create_profile

NG 766 or later: This function also works in Command Center (Metaconsole).

Required parameters:

- < profile_name >
- < rights_list > Use 1 or 0 for otorgate or deny right, respectively. They are numbered so:
  1. AR View agents.
  2. AW Edit agents.
  3. AD Disable agents.
  4. LW Edit alerts.
  5. LM Manage alerts.
  6. UM Manage users.
  7. DM Manage database.
  8. ER View events.
  9. EW Edit events.
  10. EM Manage events.
  11. RR View reports.
  12. RW Edit reports.
  13. RM Manage reports.
  14. MR View network maps.
  15. MW Edit network maps.
  16. MM Manage network maps.
  17. VR View visual console.
  18. VW Edit visual console.
  19. VM Manage visual console.
  20. PM Pandora FMS management.
  21. NR View NCM data.
  22. NW Operate NCM -you must include NR-.
  23. NM Manage NCM -you must include NW and NR-.

Descripción:

A new profile will be created by passing the profile name and permissions as parameters. **All fields are required.**

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --create_profile Profile01 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

update_profile

NG 766 or later: This function also works in Command Center (Metaconsole).

Required parameters:

- < profile_name >
- < rights_list > Use 1 or 0 for otorgate or deny right, respectively. They are numbered so:
    1. AR View agents.
    2. AW Edit agents.
    3. AD Disable agents.
    4. LW Edit alerts.
    5. LM Manage alerts.
    6. UM Manage users.
    7. DM Manage database.
    8. ER View events.
    9. EW Edit events.
    10. EM Manage events.
    11. RR View reports.
    12. RW Edit reports.
    13. RM Manage reports.
    14. MR View network maps.
    15. MW Edit network maps.
    16. MM Manage network maps.
    17. VR View visual console.
    18. VW Edit visual console.
    19. VM Manage visual console.
    20. PM Pandora FMS management.
    21. NR View NCM data.
    22. NW Operate NCM -you must include NR-.
    23. NM Manage NCM -you must include NW and NR-.

Descripción:

A existent profile will be updated by passing the profile name and permissions as parameters. **All fields are required.**

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --update_profile Profile01 1 0 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
```

add_profile

NG 766 or later: This function also works in Command Center (Metaconsole).

Required parameters:

- < id_user >
- < profile_name >
- < group_name >

Description:

A profile will be added to an user giving it as parameter the names of user, profile an group on which they will have the privileges of this profile. You should specify the group `All` if you want that the profile has validity on all groups.

Example (see call and syntax):

```
pandora_manage.pl /etc/pandora/pandora_server.conf --add_profile usuario002
'Group coordinator' All
```

add_tag_to_user_profile

Parameters: <user_id> <tag_name> <group_name> <profile_name>

Description: Add a tag to the given user profile.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
add_tag_to_user_profile '7' 'new tag' 'All' 'Profile for user 7'
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

delete_profile

NG 766 or later: This function also works in Command Center (Metaconsole).

Required parameters:

- < id_user >
- < profile_name >
- < group_name >

Description:

An user profile will be deleted giving it as parameter the names of user, profile and group on which the profiles has the priviledges. If the profile to delete is associated to the `All` group, we should specify as group `All`.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --delete_profile user02 'Chief
Operator' Applications
```

add_profile_to_user

NG 766 or later: This function also works in Command
Center (Metaconsole).

Required parameters:

- < id_user >
- < profile_name >

Optinal parameter:

- < group_name >

Description:

Add a existing profile in group to a user. If the group is not provided, the grupo will be `All`.

Example (see call and syntax):

```
pandora_manage.pl /etc/pandora/pandora_server.conf --add_profile_to_user
'user01' 'Chief Operator' 'Network'
```

disable_eacl

Parameters: Nothing.

Description: The Enterprise mode ACL system will be disabled in the configuration with the execution of this option.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --disable_eacl
```

enable_eacl

Parameters: Nothing.

Description: The Enterprise mode ACL system will be enabled in the configuration with the execution of this option.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --enable_eacl
```

disable_double_auth

Required parameter: <id_user>

Description: The double authentication will be disabled for the specified user.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --disable_double_auth
'admin'
```

Events

create_event

Required parameters:

- Event name.
- Event type. You can take any of these values:
  - `unknown` , `alert_fired` , `alert_recovered` , `alert_ceased` , `alert_manual_validation` , `recon_host_detected` , `system` , `error` , `new_agent` , `going_up_warning` , `going_up_critical` , `going_down_normal` , `going_down_warning`, `going_down_critical` , `going_up_normal` , `configuration_change` .
- Group name.

Optinal parameters:

- Agent name.
- Module name.
- Event state (`0` if it isn't validated, 1 if is validated and 2 in process).
- Severity: `0` (Maintenance), 1 (Informational), 2 (Normal), 3 (Warning), 4 (Critical), 5 (Minor) y 6 (Major).
- Template name in the case that is would be associated to one alert.
- User name.
- Comment.
- Source.
- Extra id.
- Tags: Format should be < tag > < url >,< tag > < url > You can add multiple tags separated by commas. *It is important that there are no spaces between the comma and the next label.*
- Custom data: Custom data should be entered as a JSON document. For example: `{“Location”: “Office”, “Priority”: 42}` .

- Force creation of agent (*bool*): If the agent name parameter refers to an agent that does not exist, it will be created.
- Critical instructions: Add the instructions to be performed in the case of a critical condition
- Warning instructions: Add the instructions to be performed in the case of a warning condition
- Unknown instructions: Add the instructions to perform in case of going to unknown state
- Usar alias del agente, utilice textualmente `use_alias`. *Se deben especificar dos comillas simples juntas, sin espacio entre ellas) si utiliza el siguiente parámetro a continuación*.
- Command Center (Metaconsole): Server identifier.

Description:

An event will be created with these data: the name and kind of the event, name of the module, agent and group associated.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --create_event \
    'Event CLI' \
    system \
    Firewalls \
    'My agent' 'My module' 0 4 Template004 \
```

Validate_event

Parameters: <agent_name> <module_name> <datehour_min> <datehour_max> <name_user> <criticity> <template_name> <use_agent_alias>

Description: All events will be validated considering a group of filters. The configurable filters are: the agent name, the module name, date-hour minimum and date-hour maximum, the user name, the severity and the name of the associated template.

It's possible to combine the parameters in several ways, leaving blank with empty inverted commas (*) the ones that you don't want to use and filling in the rest. Example: perl pandora_manage.pl /etc/pandora/pandora_server.conf –validate_event 'My agent' 'My module'* '2010-06-02 22:02'

In this example will be validated all the events associated to the module 'Mi module' of the agent 'My agent' which data would be previous to 2 june 2010 not considering the rest of the filters. It would be also possible to filter the events between two dates filling both of them or the ones that have a data higher to an specific one, filling in only the date-hour minimum.

Validate_event_id

(>=5.0)

Parameters: <id_event>

Description: A event will be validated.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --validate_event_id 1894
```

In this example, will be validated the event whose identifier is 1894.

Get_event_info

(>=5.0)

Parameters: <id_event>[<separator]

Description: Display info about a event given a id.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_event_info 1894
```

In this example, will be displayed info about the event whose identifier is 1894. The fields will be separated by |

add_event_comment

Required parameters:

- < id_event >
- < id_user >
- < comment >

Description:

Add a comment to an event.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --add_event_comment 1234 admin
TEXT
```

In this example, comment TEXT by admin will be added to the event whose identifier is 1234.

event_in_progress

Parameters: <id_event>

Description: Set event in progress.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --event_in_progress 123
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

update_event_custom_id

Required parameters:

- < id_event >
- < id_custom_event >

Description:

It adds a custom identifier to an event.

Example (see call and syntax):

```
pandora_manage /etc/pandora/pandora_server.conf --update_event_custom_id '232'
'CLI test'
```

Policies

Create_policy

Parameters: <policy_name> <group> <description>

Description: A policy will be created with policy name, group and description. The description is not mandatory.

Example:

```

```

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_policy 'My
policy' 'My group' 'Description'
```

Apply_policy

Parameters: <policy_name>

Description:A policy passed as parameter will be forcibly applied. Within the process of applying a policy is: the creation of the modules of the policy in all its associated agents, the creation of the alerts of the policy in the created modules, and the accomplishment of changes in the configuration file of the local agents that the policy may have to add the created modules and the collections associated to the policy.

As this command works through the API, it needs the following parameters:

- id =<id_policy> (required)
- id2=<id_agent> (optional) Id or name of the agent as indicated in the other parameter. If it is empty, the policy will be applied to all the agents of this.
- other =<serialized parameters>
  - <name_agent (Boolean)> Indicates if the agent will be sent by Id (0) or by name (1).
  - <server_id> (required if using Command Center -Metaconsole-) Id of the server in which the policy will be applied.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --apply_policy
'policy_id' 'agent_id' 'other1|other2'
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

Apply_all_policies

(>=5.0)

Parameters: None

Description: Add to the application queue all the policies. The server is who watch the queue and apply the policies

Example:

```

```

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --apply_all_policies
```

Add_agent_to_policy

(>=5.0)

Parameters: <agent_name> <policy_name> <use_agent_alias>

Description: An existent group will be added to an existent policy

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --add_agent_to_policy
'Agent name' 'Policy name'
```

Add_collection_to_policy

Parameters: <policy_name> <collection_name>

Description: An existent collection will be added to an existent policy

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
add_collection_to_policy 'Policy name' 'Collection name'
```

recreate_collection

Parameters: <collection_id>

Description: Recreate the files of a collection. Returns `[INFO] Collection recreated successfully.` otherwise `[ERROR] Collection not recreated.`

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --recreate_collection 7
```

Create_policy_data_module_from_local_component

Parameters: <policy_name> <component_name>

Description: A module will be created from a local component to a policy

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
create_policy_data_module_from_local_component 'Policy name' 'Component name'
```

delete_not_policy_modules

Parameters: Nothing.

Description: All modules that doesn't belong to any policy will be deleted both from the database and the agent configuration file (if there is one).

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
delete_not_policy_modules
```

Remove_agent_from_policy

Parameters: <id_policy> <id_agent>

Description:Removes an agent from a policy. It is necessary to specify the id of the policy and the id of the agent to be deleted.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
remove_agent_from_policy '11' '2'
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

Disable_policy_alerts

Parameters: <policy_name>

Description: All the alerts from a policy passed by parameter will be flagged as disabled

Example:

```

```

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --disable_policy_alerts
'My policy'
```

Create_policy_data_module

(>=5.0)

Parameters: <policy_name> <module_name> <module_type> [<description> <module_group> <min> <max> <post_process> <interval> <warning_min> <warning_max> <critical_min> <critical_max> <history_data> <data_configuration> <warning_str> <critical_str> <enable_unknown_events> <ff_threshold> <each_ff> <ff_threshold_normal> <ff_threshold_warning> <ff_threshold_critical> <ff_timeout> <critical_instructions> <warning_instructions> <unknown_instructions>

Description: A policy data module will be created. The default values are the same of –create_data_module option

Notes:

The next parameters are only for the Pandora version 5.1 and next versions:

- <ff_threshold>
- <each_ff>
- <ff_threshold_normal>
- <ff_threshold_warning>
- <ff_threshold_critical>
- <ff_timeout>

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
create_policy_data_module 'policy name' 'module name' generic_proc 'module
description' 'group name' 0 100 0 300 30 60 61 100 0 "module_begin\nmodule_name
modname\nmodule_end" 'string for warning' 'string for critical'
```

Create_policy_web_module

(>=7.0)

Parameters: <policy_name> <module_name> <module_type> [<description> <module_group> <min> <max> <post_process> <interval> <warning_min> <warning_max> <critical_min> <critical_max> <history_data> <retries> <requests> <agent_browser_id> <auth_server> <auth_realm> <configuration_data> <proxy_url> <proxy_auth_login> <proxy_auth_password> <warning_str> <critical_str> <enable_unknown_events> <ff_threshold> <each_ff> <ff_threshold_normal> <ff_threshold_warning> <ff_threshold_critical> <ff_timeout>

<warning_inverse> <critical_inverse> <critical_instructions> <warning_instructions> <unknown_instructions>]

Description: A policy web module will be created. The default values are the same of –create_web_module option

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
create_policy_web_module 'policy_name' 'module_name' web_data
'module_description' 'General' 0 100 0 300 0 0 0 0 1 0 1 'Pandora FMS' auto
public "module_begin\nmodule_name modname\nmodule_end" 'http://proxy.url'
'proxy_login' 'proxy_password' 1 10 10 10 10 10 10 10 10 1 1
'critical_instructions' 'warninng_instructions' 'unknown_instructions'
```

Create_policy_network_module

(>=5.0)

Parameters: <policy_name> <module_name> <module_type> [<module_port> <description> <module_group> <min> <max> <post_process> <interval> <warning_min> <warning_max> <critical_min> <critical_max> <history_data> <ff_threshold> <warning_str> <critical_str> <enable_unknown_events> <each_ff> <ff_threshold_normal> <ff_threshold_warning> <ff_threshold_critical> <critical_instructions> <warning_instructions> <unknown_instructions>]

Description: A policy network module will be created. The default values are the same of –create_network_module option

Notes:

The next parameters are only for the Pandora version 5.1 and next versions:

- <each_ff>
- <ff_threshold_normal>
- <ff_threshold_warning>
- <ff_threshold_critical>

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
create_policy_network_module 'policy name' 'module name' remote_icmp_proc 22
'module description' 'group name' 0 100 0 300 30 60 61 100 0 0 'string for
warning' 'string for critical'
```

Create_policy_snmp_module

(>=5.0)

Parameters: <policy_name> <module_name> <module_type> <module_port> <version> [<community> <oid> <description> <module_group> <min> <max> <post_process> <interval> <warning_min> <warning_max> <critical_min> <critical_max> <history_data> <snmp3_priv_method> <snmp3_priv_pass> <snmp3_sec_level> <snmp3_auth_method> <snmp3_auth_user> <snmp3_priv_pass> <ff_threshold> <warning_str> <critical_str> <enable_unknown_events> <each_ff> <ff_threshold_normal> <ff_threshold_warning> <ff_threshold_critical> <critical_instructions> <warning_instructions> <unknown_instructions>]

Description: A policy SNMP module will be created. The default values are the same of –create_snmp_module option

Notes:

The next parameters are only for the Pandora version 5.1 and next versions:

- <each_ff>
- <ff_threshold_normal>
- <ff_threshold_warning>
- <ff_threshold_critical>

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
create_policy_snmp_module 'policy name' 'module name' remote_snmp_inc 8080 1
mycommunity myoid 'Module description'
```

Create_policy_plugin_module

(>=5.0)

Parameters: <policy_name> <module_name> <module_kind> <module_port> <plugin_name> <user> <password> <parameters> [<description> <module_group> <min> <max> <post_process> <interval> <warning_min> <warning_max> <critical_min> <critical_max> <history_data> <warning_str> <critical_str> <enable_unknown_events> <each_ff> <ff_threshold_normal> <ff_threshold_warning> <ff_threshold_critical> <critical_instructions> <warning_instructions> <unknown_instructions>]

Description: A policy plugin module will be created. The default values are the same of –create_plugin_module option

Notes:

The next parameters are only for the Pandora version 5.1 and next versions:

- <each_ff>
- <ff_threshold_normal>
- <ff_threshold_warning>
- <ff_threshold_critical>

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
create_policy_plugin_module 'policy name' 'module name' generic_data 22 myplugin
myuser mypass 'param1 param2 param3' 'Module description' 'General' 1 3 0 300 0
0 0 0 1 'string for warning' 'string for critical'
```

Validate_policy_alerts

(>=5.0)

Parameters: <policy_name>

Description: Validate all the alerts of a given policy

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --validate_policy_alerts
'Policy name'
```

Get_policy_modules

(>=5.0)

Parameters: <policy_name>

Description: Get the module list (id and name) of a given policy

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_policy_modules
'Policy name'
```

Get_policies

(>=5.0)

Parameters: [<agent_name> <use_agent_alias>]

Description: Get all the policies (without parameters) or the policies of a given agent (agent name

as parameter)

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_policies 'Agent
name'
```

Netflow

Create_netflow_filter

(>=5.0)

Parameters: <filter_name> <group_name> <filter> <aggregate_by> <output_format>

Description: Create a new netflow filter.

The possible values of aggregate_by parameter are: dstip,dstport,none,proto,srcip,srcport The possible values of ouput_format parameter are: kilobytes,kilobytespersecond,megabytes,megabytespersecond

Example:

To create a netflow filter we execute the following option:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_netflow_filter
"Filter name" Network "host 192.168.50.3 OR host 192.168.50.4 or HOST
192.168.50.6" dstport kilobytes
```

Tools

Exec_from_file

(>=5.0)

Parameters: <option_to_execute> <option_params>

Description: With this option is possible to execute any CLI option with macros from a CSV file. The number of macros will be the number of columns in the CSV file. Each macro will be named FIELD1 , FIELD2 , FIELD3 etc.

Example:

We are going to create users from a CSV file.

We need a CSV file like that:

```
User 1,Password 1,0
User 2,Password 2,0
User 3,Password 3,0
User Admin,Password Admin,1
```

The name of the file will be '/tmp/users_csv'

We are going to execute the option –create_user with the following options: <user_name> <user_password> <is_admin> <comments>

To do this, we execute the following option:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --exec_from_file
/tmp/users_csv create_user __FIELD1__ __FIELD2__ __FIELD3__ 'User created with
exec_from_file option from CLI'
```

NOTE: Commas into the CSV columns are not yet supported

create_snmp_trap

Parameters: <name> <oid> <desc> <severity>

Name: As seen in the snmp trap console.

OID: SNMP trap main OID.

Severity: Numeric value, which have following values: Severity 0 (Maintenance), 1(Info) , 2 (Normal), 3 (Warning), 4 (Critical), 5 (Minor) and 6 (Major).

Sample:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_snmp_trap
Cisco_FAN_Crash 1.3.3.3.2.12.3.3.4.1 "Something happen with the FAN inside the
CISCO device, probably a failure" 3
```

start_snmptrapd

Parameters: Nothing.

Starts the `snmptrapd` process on the traps server. In case it is already started, it terminates the service (kill command) and runs it again, deleting all the traps located in the exchange log (by

default `/var/log/pandora/pandora_snmptrapd.log`).

Example:

```
pandora_manage /etc/pandora/pandora_server.conf --start_snmptrapd
```

Graphs

create_custom_graph

Parameters: <name> <description> <user> <id_group> <width> <height> <events> <graph_type> <period> <modules> <separator>

Description: You can create a graph with these elements. Two or more modules must be separated with <separator> wich always must be specified, even for a single module. *All parameters are required, but some of them can be empty by singles quotes.* Their default values are:

- Width: `550`.
- Height: `210`.
- Period: `86400` (seconds).
- Events: `0`.
- Graph type: `0`.
- Id group: `0`.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_custom_graph
'My graph' 'Created by CLI' 'admin' 0 '' '' 0 2 '' '1;2;5;30' ';'
```

edit_custom_graph

Parameters: <id_graph> <name> <description> <user> <id_group> <width> <height> <events> <graph_type> <period>

Description: You can edit a graph with these values. *All parameters are required, but some of them can be empty by singles quotes*. Fields not specified keep their values.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --edit_custom_graph 12
'' 'edit graph by CLI' '' '' '' '' '' '' 25200
```

add_modules_to_graph

Parameters: <id_graph> <modules> <separator>

Description: These modules will be added to the graph. *All parameters are required.*

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --add_modules_to_graph 2
'1015,1017' ','
```

delete_modules_to_graph

Parameters: <id_graph> <modules> <separator>

Description: These modules will be removed to the graph. *All parameters are required.*

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf –delete_modules_to_graph
2 '1015,1017' ','
```

Clusters

new_cluster

Parameters: <cluster_name> <cluster_type> <description> <group_id>

Description: A cluster with the specified parameters and their associated elements will be created.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --new_cluster cluster1
AA description 12
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

add_cluster_agent

Parameters: <json_data_base64:[{“id”:5,“id_agent”:2},{“id”:5,“id_agent”:3}]>

Description: A specific agent will be added to the specified cluster. It is absolutely necessary that JSONs are encoded on base 64.

Example :

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --add_cluster_agent
W3siaWQiOjUsImlkX2FnZW50IjoyfSwKeyJpZCI6NSwiaWRfYWdlbnQiOjN9XQ ===
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

add_cluster_item (active / active)

Parameters:

JSON data with this fields:

- `name` : (string value).
- `id_cluster` : (numeric value).
- `type` : AA (means “Active/Active”).
- `critical_limit` : (numeric value).
- `warning_limit` : (numeric value).

Example JSON:

```
[
  {
    "name": "Swap_Used",
    "id_cluster": 5,
    "type": "AA",
    "critical_limit": 80,
    "warning_limit": 60
  },
  {
    "name": "TCP_Connections",
    "id_cluster": 5,
    "type": "AA",
    "critical_limit": 80,
    "warning_limit": 60
```

```
  }
]
```

Description: A specific module will be added as an item to the specified cluster.

It is absolutely necessary that JSON data is encoded on
base 64.

Example :

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --add_cluster_agent
WwogIHsKICAgICJuYW1lIjogIlN3YXBfVXNlZCIsCiAgICAiaWRfY2x1c3RlciI6IDUsCiAgICAidHlw
ZSI6ICJBQSIsCiAgICAiY3JpdGljYWxfbGltaXQiOiA4MCwKICAgICJ3YXJuaW5nX2xpbWl0IjogNjAK
ICB9LAogIHsKICAgICJuYW1lIjogIlRDUF9Db25uZWN0aW9ucyIsCiAgICAiaWRfY2x1c3RlciI6IDUs
CiAgICAidHlwZSI6ICJBQSIsCiAgICAiY3JpdGljYWxfbGltaXQiOiA4MCwKICAgICJ3YXJuaW5nX2xp
bWl0IjogNjAKICB9Cl0=
```

For it to be carried out correctly, the API configuration parameters will
need to be filled in `pandora_server.conf` (configuration file elements)
: `console_api_pass` , `console_pass` , `console_user` y
`console_api_url` .

add_cluster_item (active / pasive)

Parameters:

JSON data with this fields:

- `name` : (string value).
- `id_cluster` : (numeric value).
- `type` : AP (means “Active/Pasive”).
- `critical_limit` : (numeric value).
- `warning_limit` : (numeric value).

Example JSON:

```
[
  {
    "name": "DiskUsed_/proc/kcore",
    "id_cluster": 5,
    "type": "AP",
    "is_critical": 1
  },
  {
```

```
    "name": "DiskUsed_/proc/sched_debug",
    "id_cluster": 5,
    "type": "AP",
    "is_critical": 1
  }
]
```

Description: A specific module will be added as an item to the specified cluster.

It is absolutely necessary that JSON data is encoded on
base 64.

Example :

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --add_cluster_item
WwogIHsKICAgICJuYW1lIjogIkRpc2tVc2VkXy9wcm9jL2tjb3JlIiwKICAgICJpZF9jbHVzdGVyIjog
NSwKICAgICJ0eXBlIjogIkFQIiwKICAgICJpc19jcml0aWNhbCI6IDEKICB9LAogIHsKICAgICJuYW1l
IjogIkRpc2tVc2VkXy9wcm9jL3NjaGVkX2RlYnVnIiwKICAgICJpZF9jbHVzdGVyIjogNSwKICAgICJ0
eXBlIjogIkFQIiwKICAgICJpc19jcml0aWNhbCI6IDEKICB9Cl0=
```

For it to be carried out correctly, the API configuration parameters will
need to be filled in `pandora_server.conf` (configuration file elements)
: `console_api_pass` , `console_pass` , `console_user` y
`console_api_url` .

get_cluster_status

Parameters: <id_cluster>

Description: Getting cluster status.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --get_cluster_status 7
```

For it to be carried out correctly, the API configuration parameters will
need to be filled in `pandora_server.conf` (configuration file elements)
: `console_api_pass` , `console_pass` , `console_user` y
`console_api_url` .

delete_cluster

Parameters: <cluster_id>

Description: A cluster will be deleted.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --delete_cluster 1
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

delete_cluster_agent

Parameters: <agent_id> <cluster_id>

Description: An agent added to a cluster will be disassociated

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --delete_cluster_agent 1
1
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

delete_cluster_item

Parameters: <item_id>

Description: A cluster item will be deleted

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --delete_cluster_item 1
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

Visual Consoles

create_visual_console

Parameters: <name> <background> <width> <height> <group> <mode> [<position_to_locate_elemens|<background_color> <elements>]]

Description: A visual console will be created with the specified parameters and their associated elements. It is absolutely necessary that the JSONs go in single quotes because if they don't, there could be errors in the shell from which the script is launched.

“Interesting data:” The mode can adopt the values'static_objects' and'auto_creation', in the first way the elements will be created without taking into account the position where to place the elements, but the position described in the elements json itself will be used. However, if the mode is 'auto_creation' a coordinate tapestry ('position_to_locate_elements' field with the format described in the example below) must be set and the algorithm will automatically establish the position and size of each of the elements.

Example (auto_creation):

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_visual_console
'test console' 'mapamundo.jpg' 1024 768 12 'auto_creation'
'{"pos1x":0,"pos1y":60,"pos2x":800,"pos2y":600}' '#FFF' '[{"image":
"_engine","height": 0,"width": 0,"label": "","type": 0,"period": 300,"id_agent":
1,"id_agente_modulo": 0,"id_group": 0,"id_layout_linked": 0,"parent_item":
0,"enable_link": 1,"id_metaconsole": 0,"id_custom_graph": 0,"border_width":
0,"border_color": "","fill_color": "","type_graph": "area","label_position":
"down"},{"image": "_printer","height": 0,"width": 0,"label": "","type":
0,"period": 300,"id_agent": 2,"id_agente_modulo": 0,"id_group":
0,"id_layout_linked": 0,"parent_item": 0,"enable_link": 1,"id_metaconsole":
0,"id_custom_graph": 0,"border_width": 0,"border_color": "","fill_color":
"","type_graph": "area","label_position": "down"}]'
```

Example (static_objects):

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_visual_console
'test console' 'mapamundo.jpg' 1024 768 12 'static_objects' "" '#FFF'
'[{"image": "_engine","pos_x":100,"pos_y":100,"height": 0,"width": 0,"label":
```

```
"","type": 0,"period": 300,"id_agent": 1,"id_agente_modulo": 0,"id_group":
0,"id_layout_linked": 0,"parent_item": 0,"enable_link": 1,"id_metaconsole":
0,"id_custom_graph": 0,"border_width": 0,"border_color": "","fill_color":
"","type_graph": "area","label_position": "down"},{"image":
"_printer","pos_x":400,"pos_y":100,"height": 0,"width": 0,"label": "","type":
0,"period": 300,"id_agent": 2,"id_agente_modulo": 0,"id_group":
0,"id_layout_linked": 0,"parent_item": 0,"enable_link": 1,"id_metaconsole":
0,"id_custom_graph": 0,"border_width": 0,"border_color": "","fill_color":
"","type_graph": "area","label_position": "down"}]'
```

edit_visual_console

Parameters: <id> [<name> <background> <width> <height> <group> <mode> <position_to_locate_elemens> <background_color> <elements>]

Description: A visual console with the specified parameters and its associated elements will be edited. It is absolutely necessary that the JSONs go between single quotes since if it didn't occur it could cause errors in the shell from where the script is launched.

Example (auto_creation):

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --edit_visual_console 1
'test console edited'     'auto_creation'
{"pos1x":0,"pos1y":60,"pos2x":800,"pos2y":600}  '[{"id":103,"image":
"_engine","height": 180,"width": 200,"label": "","type": 0,"period":
300,"id_agent": 3,"id_agente_modulo": 0,"id_group": 0,"id_layout_linked":
0,"parent_item": 0,"enable_link": 1,"id_metaconsole": 0,"id_custom_graph":
0,"border_width": 0,"border_color": "","fill_color": "","type_graph":
"area","label_position": "down"},{"pos_x":400,"pos_y":100,"image":
"_printer","height": 0,"width": 0,"label": "","type": 0,"period":
300,"id_agent": 4,"id_agente_modulo": 0,"id_group": 0,"id_layout_linked":
0,"parent_item": 0,"enable_link": 1,"id_metaconsole": 0,"id_custom_graph":
0,"border_width": 0,"border_color": "","fill_color": "","type_graph":
"area","label_position": "down"}]'
```

Example (static_objects):

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --edit_visual_console 1
'test console edited'     'static_objects'   '[{"id":103,"image":
"_engine","height": 180,"width": 200,"label": "","type": 0,"period":
300,"id_agent": 3,"id_agente_modulo": 0,"id_group": 0,"id_layout_linked":
0,"parent_item": 0,"enable_link": 1,"id_metaconsole": 0,"id_custom_graph":
0,"border_width": 0,"border_color": "","fill_color": "","type_graph":
"area","label_position": "down"},{"pos_x":400,"pos_y":100,"image":
"_printer","height": 0,"width": 0,"label": "","type": 0,"period":
300,"id_agent": 4,"id_agente_modulo": 0,"id_group": 0,"id_layout_linked":
0,"parent_item": 0,"enable_link": 1,"id_metaconsole": 0,"id_custom_graph":
0,"border_width": 0,"border_color": "","fill_color": "","type_graph":
"area","label_position": "down"}]'
```

delete_visual_console

Parameters: <id>

Description: The specified visual console and its associated elements will be removed.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --delete_visual_console
1
```

delete_visual_console_objects

Parameters: <id> <mode> <id_mode>

Description: Items associated with a visual console will be removed by following a filter. The available modes are “type”, “image”, “id_agent”, “id_agente_modulo”, “id_group” and “type_graph” where its identifier must be specified, which may be the type ID, the image name, the agent ID, the module ID, the group ID or the type of graphic. (area, line...).

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
delete_visual_console_objects 1 'id_agent' 1
```

duplicate_visual_console

Parameters: <id> <times> [<prefix]

Description: The indicated visual console will be duplicated as many times as desired.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
duplicate_visual_console 1 10 'test console 2'
```

export_json_visual_console

Parameters: <id> [<path> <with_element_id>]

Description: Exports to json a visual console and its elements being able to specify the path where to save the file and if you want to include the id of the associated elements to facilitate a later

edition.

Interesting data: The field “with_element_id” can return 1 and 0, which mean with or without ID respectively.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --
export_json_visual_console 1 '/tmp' 1
```

Others

locate_agent

Parameters: <agent_name> [`use_alias`] .

Description: Return the id of the node that manages the agent.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --locate_agent NAS
```

create_tag

Parameters: <tag_name> <tag_description> [<tag_url>] [<tag_email>]

Description: Create a new tag. Tag name and tag description are obligatory.

Example:

```
perl pandora_manage.pl /etc/pandora/pandora_server.conf --create_tag 'New tag
name' 'Your description'
```

For it to be carried out correctly, the API configuration parameters will need to be filled in `pandora_server.conf` (configuration file elements) : `console_api_pass` , `console_pass` , `console_user` y `console_api_url` .

Go back to Pandora FMS documentation index