



**Pandora FMS 1.3.1  
User documentation**

April, 2008

## Index

1 . Introduction to Pandora FMS .....	6
1.1. What is Pandora FMS? .....	6
1.2. What else can Pandora FMS do? .....	6
1.2.1. Using agents .....	7
1.3. Pandora FMS Architecture .....	9
1.4. Pandora FMS Agents.....	10
1.4.1. Satellite Agents .....	12
1.4.2. XML data file .....	12
1.5. Pandora FMS Servers.....	14
1.6. Pandora FMS console.....	15
1.7. Database.....	15
1.7.1. Compacting data .....	15
1.8. About Pandora FMS.....	16
2. Pandora FMS Server installation .....	17
2.1. Prerequisites .....	17
2.1.1. Pandora FMS Data Server .....	17
2.1.2. Pandora FMS Network Server .....	17
2.1.3. SNMP trap reception Console .....	18
2.1.4. Network Reconnaissance Server (Recon) .....	18
2.1.5. Installing dependencies .....	18
2.2. Automatic installation using installer.....	19
2.3. Manual installation of Pandora FMS Servers.....	20
2.3.1. To create user "pandora" and set up permissions .....	20
2.3.2. Installing executables and libraries on your system .....	21
2.3.3. Setting up snmptrapd for Pandora FMS SNMP Console .....	22
2.3.4. Testing Pandora FMS Server installation .....	22
2.3.5. Testing Pandora FMS trap reception Console .....	22
2.4. Configuring Pandora FMS Server .....	23
2.4.1. New options introduced in 1.3.1 version .....	25
2.4.2. Configuring Tentacle to receive data packages .....	26
2.4.3. Configuring Tentacle Server to use it with Pandora FMS Data Server .....	27
2.4.4. Setting up SSH configuration to receive data packets .....	27
3. Installing Pandora FMS Console .....	28
3.1. Installing Pandora FMS Database.....	28
3.2. Pandora FMS Web Console installation.....	29
4. Installing and using Pandora FMS physical agents .....	31
4.1. Understanding what is a Pandora FMS Agent.....	31
4.1.1. Generic role of the agents .....	31
4.2. Pandora FMS Agent configuration.....	32
4.2.1. Main program .....	32
4.2.2. Configuration file .....	32
4.2.3. General parameters .....	32
4.2.4. Module definition .....	34
4.2.5. Examples .....	36
4.2.6. Types of Agents .....	37
4.3. Introduction to UNIX Agents.....	37
4.4. Installing Pandora FMS UNIX Agent.....	38

4.4.1. First running of the UNIX agent .....	39
4.4.2. Advanced configuration for UNIX Agent .....	39
4.5. Pandora FMS Windows Agents.....	41
4.5.1. Build Pandora FMS Windows Agent from sources .....	41
4.5.2. Installing Pandora FMS Windows Agent (installer) .....	41
4.5.3. Manual Pandora FMS Windows Agent installation .....	41
4.5.4. Windows Agent testing .....	42
4.5.5. Pandora FMS Windows Agent configuration .....	42
4.5.6. Agent visualizaton in Pandora FMS Web Console .....	44
4.6. Data transfer to server.....	44
4.6.1. Data transfer using Tentacle .....	44
4.6.2. Data transfer using SSH .....	45
4.6.3. SSH Key generation for Windows .....	47
4.6.4. Data transfer using FTP .....	51
5. Basic Monitoring .....	52
5.1. Pandora FMS Servers.....	52
5.2. Network Server .....	53
5.2.1. Network Modules .....	53
5.2.2. Pandora FMS and SNMP .....	54
5.2.3. Network profiles .....	58
5.3. Recon Server .....	59
5.3.1. Example of use .....	60
5.4. Pandora FMS Agents.....	62
5.4.1. Pandora FMS physical agents .....	62
5.4.2. Network agents / Network modules .....	63
5.4.3. Assigning modules to a Pandora FMS agent .....	63
5.4.4. Creating a logic agent from a network module .....	65
5.5. Viewing data with Pandora FMS .....	71
5.5.1. Tactical view .....	71
5.5.2. Group view .....	72
5.5.3. Agent global detail .....	73
5.5.4. Details of Pandora FMS logic agent .....	75
5.5.5. Monitor detai .....	75
5.5.6. Alert detail .....	76
5.5.7. Data detail .....	76
5.5.8. Exporting data .....	78
5.6. Basic incident managemen .....	79
5.6.1. Adding an incident .....	80
5.6.2. Tracking an inciden .....	80
5.6.3. Adding notes to an event .....	80
5.6.4. Adding files to an event .....	80
5.6.5. Creating an indicent from an event .....	81
5.6.6. Autogenerated incidents (Recon Server) .....	82
5.6.7. Searching incidents .....	82
5.6.8. Statistics .....	82
5.7. Users in Pandora FMS .....	82
5.7.1. Profiles .....	83
6. Advanced Monitoring .....	84

6.1. Alerts in Pandora FMS .....	84
6.1.1. Adding and editing alerts .....	84
6.1.2. Alert definition examples .....	85
6.1.3. Assigning Alerts to modules .....	87
6.1.4. Pandora FMS alert log .....	89
6.2. User graphics.....	89
6.3. Creating custom reports .....	91
6.4. Setting up a Visual Console .....	93
6.5. Massive copy/deletion/propagation of Modules and Alerts .....	95
7. Maintenance and Tools .....	97
7.1. Pandora FMS maintenance .....	97
7.1.1. Installing cron job for Pandora FMS DB maintenance tool .....	98
7.2. Pandora FMS DB Stress tool .....	98
7.2.1. Pandora FMS DB Stress tool fine tune .....	99
7.3. Database maintenance .....	99
7.3.1. Manual purge of the Database .....	100
7.3.2. Normalizing data .....	101
7.3.3. Database backup .....	101
8. Other advanced topics .....	103
8.1. Pandora FMS Data Server virtual servers .....	103
8.2. Pandora FMS Database design.....	103
8.2.1. Database index and other technical improvements .....	105
8.3. MySQL optimization for enterprise grade systems.....	105
8.3.1. General advices .....	105
8.3.2. Server setup variables .....	106
8.4. NTP update.....	107
8.5. SSH server securization.....	107
8.5.1. What is sponly? .....	107
8.6. FTP Server hardening (ProFtpd).....	108
8.7. vsftpd hardening.....	109
8.8. Integrating alerts with Jabber IM.....	109
8.8.1. Installing Jabber services .....	109
8.8.2. More examples of Jabber usage .....	110
8.9. Using Image_Graph (PEAR) with Pandora FMS.....	110
8.9.1. Introduction .....	111
8.9.2. Installation .....	111
8.9.3. Problems with Pear Image_Graph .....	111
8.9.4. Problems with PHP Safemode .....	112
8.9.5. Interesting links .....	112
8.10. SMS Gateway.....	112
8.10.1. SMS Gateway implementation .....	113
8.10.2. HP Network Node Manager (NNM) integration with Pandora FMS (SNMP) .....	116
8.10.3. Using the alert on a module .....	121
8.10.4. Visualizing data in NNM .....	122
9. High Availability .....	124
9.1. PandoraFMS's High Availability Features.....	124
9.1.1. MySQL Cluster .....	125

9.1.2. Multiple PandoraFMS's consoles .....	125
9.1.3. HA over PandoraFMS data server .....	125
9.1.4. HA over PandoraFMS network server .....	127
9.2. HA and load balancing with LVS and Keepalived .....	128
9.2.1. What to do when a node fails .....	129
9.3. Appendix 1. LVS load balancer configuration.....	129
9.4. KeepAlived configuration.....	130
10. What is new in Pandora FMS 1.3 .....	131
10.1. Pandora FMS 1.3.....	131
10.2. Pandora FMS 1.3.1 .....	132
10.2.1. New documentation .....	132
10.2.2. Tentacle .....	132
10.2.3. New server options .....	133
10.2.4. Other new features .....	133
10.2.5. Bugs fixed .....	134

# 1 INTRODUCTION TO PANDORA FMS

---

## 1.1 WHAT IS PANDORA FMS?

---

**Pandora FMS** is a monitoring application to watch systems and applications. Pandora FMS allows to know the status of any element of your bussiness systems. Pandora FMS watches your hardware, your software, your multilayer system and, of course, your Operating System. Pandora FMS can detect a network interface down or the movement of any *NASDAQ new technology market* value. If you wish, Pandora FMS can send a SMS message when your system or your application fails... or when Google stock value drops below 330 US\$.

Pandora FMS will adjust, like an octopus, to your systems and requirements, because it has been designed to be open, modular, multiplatform and easy to customize. Pandora FMS is developed for system administrators.



Figure 1: Pandora FMS

## 1.2 WHAT ELSE CAN PANDORA FMS DO?

---

Pandora FMS is a monitoring tool that not only measures if a parameter is right or wrong. Pandora FMS can quantify the state (right or wrong), or store a value (numeric or alphanumeric) for months if necessary. Pandora FMS can measure performances, compare values among differen systems and set alarms over thresholds. Pandora FMS works against a Database so that it can generate reports, statistics, SLA and measure anything: Operating Systems, aplications and hardware systems such as firewalls, proxies, Databases, web servers, VPN, routers, switches, processes, services, remote accesses to servers, etc. everything integrated in a open and distributed architecture. Pandora FMS can be deployed over any Operating System, with specific agents for each platform. There are already agents for Windows (2000, XP, 2003), GNU/Linux, Solaris, HP-UX, BSD, AIX, IPSO and OpenWRT.

Pandora FMS not only gathers information through its agents, but it can also monitor any hardware sistem with TCP/IP connectivity such as load balancing systems, routers, switches, printers, etc. through SNMP and TCP/ICMP checks.

Often the question "What kind of things can be monitored?" shows up, since Pandora FMS can virtually monitor anything, sometimes is convenient give some specific examples. Pandora FMS can monitor any process or system that, though a command, returns a value, and also any value inside a log file of the Operating System. Some examples of deployments already in use are:

### 1.2.1 Using agents

- Number of Checkpoint FW-1 connections (sessions).
- Number of Checkpoint FW-1 NAT sessions.
- Number of firewall connections for GNU/Linux (NetFilter/IPTables).
- Number of packets registered in Checkpoint FW-1.
- Number of packets discarded in Checkpoint FW-1.
- Number of packets accepted in Checkpoint FW-1.
- Status of HA in FW1 NG.
- Last installed policy in a Firewall-1 module.
- Status of sincronization of FW1 NG modules.
- System CPU: idle, user and system.
- Number of system processes.
- CPU temperature.
- Value of a Windows registry.
- Queued processes at a generic dispatcher.
- System memory: free, swap, kernel FW-1, cache, etc.
- Percentage of free space in disk (each partition).
- Messages processed by a mail gateway.
- Strings inside a text file.
- IP traffic (filtered by the firewall connections).
- Web page hits in several web servers (Apache, iPlanet, IIS, etc.).
- Percentage of erroneous packets in a gateway.
- Stablished connections in a RAS.
- Size of a file.
- Opened VNP server sessions.
- MySQL performance: queries and statements per second, cache leved used, cache hit rate, slow queries and simultaneous sessions.
- Status of Snort systems (events per second, sensor stats, loaded policies, etc).
- Events provided by IDS (Snort) up to six priority levels or by groups.
- Number of local connections (TCP, UPD, UNIX sockets) and detailed statistics of the OS network layer (packets fragmentation, loss, martian packets, and many other anomalies detected by the kernel).
- Viruses detected by an anti-virus gateway.
- ICMP latency times to a host.
- Average transfer ratio in a file transfer tool.
- Number of attended DNS requests by a server (including types).
- Number of attended FTP sessions by a FTP server.
- (Generic) Status of any active process/service in the system.
- (Generic) Status of any quantifiable system parameter.

**Without agents (remote monitoring)**

- Whether a system replies to pings or not (system alive).
- Know the latency time of a system (in milliseconds).
- Whether a remote TCP port is open or not.
- Know the status of a remote TCP system depending on a replied string.
  - i.e. we can know if the SSH version of a remote system is active and hasn't changed.
  - i.e. we could also verify that a web page hasn't been modified and that is working correctly.
- Get information using SNMP.
- Whether a remote UPD port is active.



## 1.3 PANDORA FMS ARCHITECTURE

Pandora FMS is extremely modular and decentralized. The most important component, and where everything is stored is the Database (right now only MySQL is supported). **Every single component of Pandora FMS** can be replicated and work under a pure HA system (Active/Passive) or under a clusterized system (Active/Active with balanced load).

- **Web Console:** Pandora FMS's user interface. The user controls and operates the

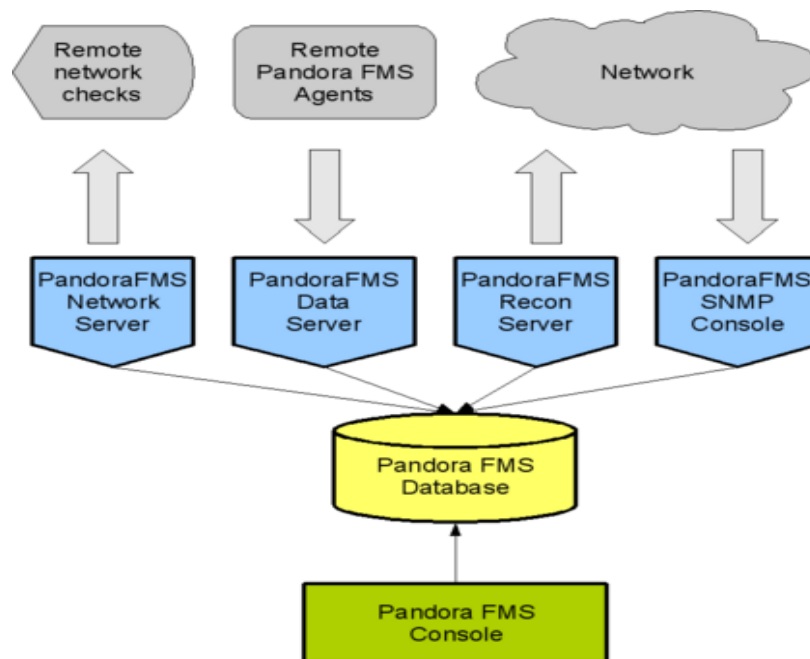


Figure 2: Global architecture

system with it. Several Web consoles can be implemented in a single system. The Web console is written in PHP, and it lies on top of a database and a Web server. It is compatible with any platform GNU/Linux, Solaris, Win2000, AIX, etc. The official supported platform is GNU/Linux, though The console allows the user to control the status of the agents, view statistical information, generate graphs and data tables and also keeps a system incident control. Moreover, it is able to generate reports and change the alerts, agents, and user profile settings.

- **Pandora FMS servers:** In Pandora FMS 1.3.1 there are four different kind of servers:
  1. The **Core Server** (called **Data Server**) is the receiver of the data packages generated by the agents and it also processes them.
  2. The **Network Server** monitorizes remote systems using network resources like ICMP, TCP, UDP or SNMP Queries. Network Servers act themselves as "Network Agents" and they gather all the data remotely.
  3. **Recon Server**, that sweeps networks detecting new systems on it and adding them to Pandora FMS so that it can monitor them automatically.
  4. The **SNMP Console**, receives and processes SNMP traps and sets the associated alerts.
- **Database:** The central Database saves all the information needed by Pandora FMS to work.
- **Pandora FMS's Agents:** they gather all the system data. They are executed in each local system. They have been developed to work under a specific platform, using specific tools of each host system.

## 1.4 PANDORA FMS AGENTS

---

Pandora FMS agents are based on native languages for every platform: shellscripts for UNIX, including GNU/Linux, Solaris, AIX, HP-UX, BSD and also Nokia's IPSO. It's possible to reproduce any agent in any programming language, however it must be compliant with Pandora FMS server's data transfer API, defined in XML, and it must be *Open Source*. Windows agents are developed under a free C++ framework (Mingw) and they use the same interface and modularity as the UNIX agents.

These scripts are formed by modules, each one gathering a "chunk" of information. Thus, every agent gathers several "chunks" of information. this one is organized in a data set and stored in a single file, called data file.

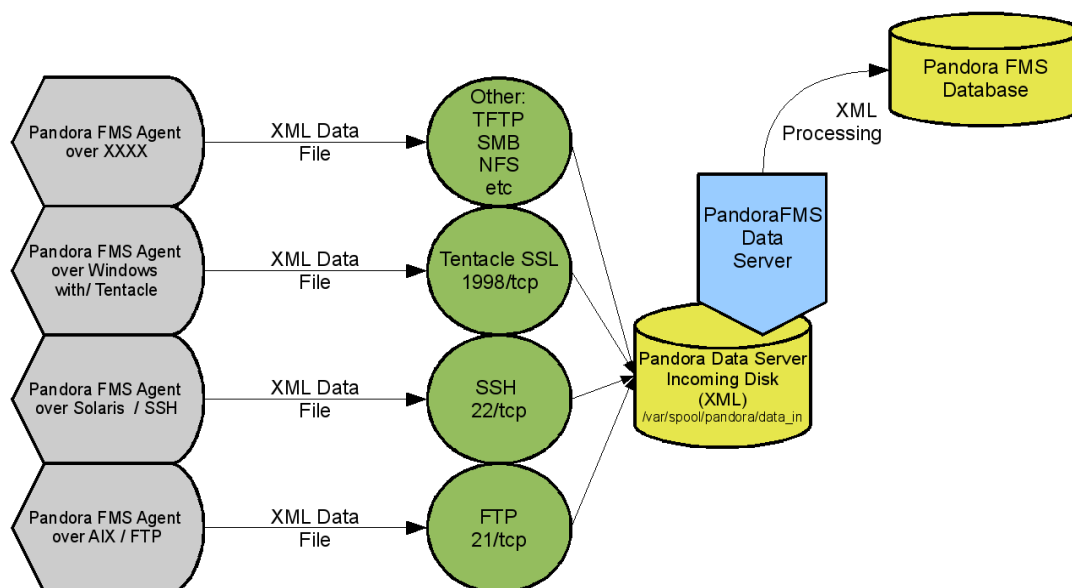


Figure 3: Pandora FMS data gathering

The process of transferring the data file from the agent to the server is made regularly at a defined time interval inside the agent configuration file, *pandora\_agent.conf*. It's possible to modify that parameter in order not to fill the database with non-relevant information, either load the network or affect the system performance. The default interval is 300 (seconds), which is equivalent to five minutes. Values under 100 (seconds) are not recommended because host performance could be affected. Besides, this causes a high load of the Database and Operating System of Pandora FMS Server. Pandora FMS is not a real time system; it is a general Monitoring System for applications and systems for environments where real time is not a critical issue.

Packets transfers are usually made through SSH, with DSA authentication (although also RSA can be used). With Pandora FMS 1.3.1 we introduced the protocol **Tentacle** [1] to transfer data files from the agents to the server much more easily.

The process is completely safe since neither any password nor unencrypted confidential information is sent. Confidentiality, integrity and authentication of the connections between the agent and the server are ensured. In the Agents and Server Installation and Configuration guides, the process of generation of keys to do the automatic SCP transfer is detailed.

Either with SSH or Tentacle the transfers are completely safe since passwords and confidential data do not travel over the net, and the connection, integrity, authentication and confidentiality between the sever and the agent are secured. The key generation process and the installation and configuration for automatic data transfer with both SCP (SSH) and Tentacle are documented.

FTP or any other file transfer system could also be used, although SSH was chosen for security and compatibility with most of the systems in the market.

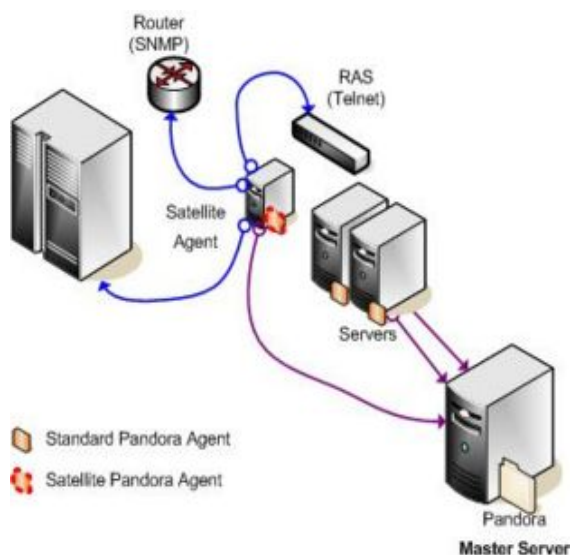
Pandora FMS Agents are thought to be executed from the agent that they gather

information with, although the agents can gather information of any reachable machine from the host where they are installed too. In this case those agents are called "Satellite Agents".

### 1.4.1 Satellite Agents

A satellite agent is the one that monitors other host elements using commands that gather remote information. Therefore, we can have a group of agents in a system, or only one that makes tests over different remote systems and integrates them as different modules inside the same agent.

A machine can also be configured to have several Pandora FMS Agents at the same time, this is quite usual, when there is a satellite agent and a "normal" agent. The standard agent monitors the machine where it is executed, while the installed satellite agents (it can have several) monitor remote systems, using Telnet, SNMP or other proprietary commands.



It is also possible to have a host with several agents. Some of them gather information from reachable machines (acting like "satellite agents") and other as estandar agents that monitors the host where they are being executed.

### 1.4.2 XML data file

The data file has the following syntax:

```
<hostname>.<serialnumber>.data
```

This is an XML file, and its name is the combination of the hostname where the agent is excued, a different serial number for every data package and the extension *.data* which indicates that it is a data file.

We also have a control file for every data file:

```
<hostname>.<serialnumber>.checksum
```

This file has a *.checksum* extension and contains a MD5 hash of the data file. This allows checking that the information has not been changed before being processed.

The XML data file generated by every agent is the core of Pandora FMS. This file has the information gathered by the Agent. It is an easy structure which allows any user to create its own developments to be processed in Pandora FMS, or use the included ones. The datafile has a similar XML sintax as the following:

```
<agent data os_name="SunOS" os_version="5.8" timestamp="300"
agent_name="pdges01" version="1.0">
```

```
<module>
<name>SSH Daemon</name>
<type>generic_proc</type>
<data>1</data>
</module>
<module>
<name>FTP Daemon</name>
<type>generic_proc</type>
<data>0</data>
</module>
<module>
<name>DiskFree</name>
<type>generic_data</type>
<data>5200000</data>
</module>
<module>
<name>UsersConnected</name>
<type>generic_data_inc</type>
<data>119</data>
</module>
<module>
<name>LastLogin</name>
<type>generic_data_string</type>
<data>slerena</data>
</module>
</agent_data>
```

## 1.5 PANDORA FMS SERVERS

---

There are four different kind of servers in Pandora FMS 1.3.1 version:

- **Pandora Data Server.** This is a PERL application that processes the information sent by the agents. The agents send the XML data file (via FTP, SSH or Tentacle) and the server periodically verifies if it has new data files waiting to be processed. You can setup different data servers in different systems or in the same host (that will be different virtual servers). Several servers can work together to monitor a big system. Pandora FMS Database, which stores all the data packets and that is shared with the Web Server, is accessed by the Data Server to get the information. The server is executed as a daemon or as a service, and processes the packets stored in its filesystem. In spite of its simplicity and its scarce use of resources, the Data Server is one of the critical elements of the system, since it processes all the information gathered by the agents, and it generates the alerts and events of the system based on that data.
- **Pandora Network Server.** This is a PERL application that executes network tasks, like sending pings, TCP requests, SNMP requests and UDP request. When you assign an agent to a server, you are assigning it to a Network Server, not to a Data Server, so, it is very important that machines running Network Servers have "network visibility" to hosts assigned in network modules. For example, if you create a module to make a ping check to 192.168.1.1 and assign this agent/module to a server in a 192.168.2.0/24 network without access to 192.168.1.0/24 module will always report *DOWN*.
- **Pandora FMS SNMP Console.** This is a PERL application that parses the output from standard snmptrapd (we provide a binary for snmptrapd, but it is possible that you need to replace it with a binary that runs better in your system). This daemon receives SNMP traps, and Pandora FMS SNMP Server stores those in its Database and fire the assigned assigned in Pandora FMS SNMP Console.
- **Pandora FMS Recon Server.** Used to sweep regularly our network and detect new alive systems. Recon server can also apply a monitoring "template" to new detected systems, and apply default modules in that template to be used to monitor immediatly the new system.

The Pandora FMS Server can work in **High Availability** and/or **Load Balancing**. In a very big architecture, several Pandora FMS Servers can be arranged simultaneously in order to be able to manage big volumes of information distributed by geographical or functional zones.

Pandora FMS Server is always running (as a daemon) and permanently verifying whether any element fires an alarm. If so, it executes the action defined in the alarm, like send a SMS, an email, or even activate the execution of a script or it send an HTTP form.

You can have several simultaneous servers, one of them would be the Main Server, or "Master Server", and the rest of the servers would be "Slave Servers". The Master Server is the only one that verifies the alarms, in case any agent goes down. The server that receives

the data file from the agent will always fire the rest of the alarms defined in the agents' modules. This is also important to know if this server changes (due to configurations of **High Availability**, load balancing or clustering), and monitor the status of each server. This is automatically done through the status section of the servers in the Web Console.

## 1.6 PANDORA FMS CONSOLE

---

The Web application of Pandora FMS allows to see graphical reports, the state of every agent, and also access to the information sent by the agent, see every monitored parameter and see its evolution throughout the time, from the different nodes, groups and users of the system. It is the part that interacts with the final user, and that will allow you to manage the system.

The Web Console is written in PHP and no plugin, Flash, Java or ActiveX is required to access the console, only a web browser that supports HTML and CSS (IE5+ or Mozilla 4+).

Pandora FMS Web Console can run in several servers, which means that you can have as many Web Consoles as wished, to balance the load or to ease the access, due to logistic problems (big networks, big number of different user groups, different geographical zones, different management, etc.). The only requirement is to be able to reach the data container, where Pandora FMS stores everything: the Database.

## 1.7 DATABASE

---

Pandora FMS uses a SQL Database to store all the information. Pandora FMS maintains an asynchronous database with all data received, making a temporary cohesion of everything it is received, and normalizing all the information from the different sources. Every Agent data module generates information input for every data bundle, which implies that a real production system can have around ten million of "data", or information atoms.

This data is managed automatically from Pandora FMS, carrying out a periodic and automatic maintenance of the database. This means that there is neither an operator, nor a manager, required to run tasks like Database administration. This is done through a regular purging of the data after a date (90 days by default), and also compacting data older than a configurable number of days old (30 days by default).

### 1.7.1 Compacting data

Data stored by Pandora FMS is useful to see evolution throughout the time, in order to make statistics, generate reports and to do capacity planning, as well as other statistical tasks. To do that it is not necessary to have all the data, but a representative sample, of smaller resolution, enough to carry out the task needed.

With that philosophy the compaction system has been constructed. For instance, if we have a sample of 9,000 elements, distributed along 90 days, Pandora FMS will take the data of last month, which would be 3,000 elements and will compress it in 300. In the graphs they

will practically be equal, and it will be useful for the reports, statistics and other tasks. This is made thanks to a interpolation in temporary strips, in a completely automatic and periodic way, where neither the user, nor the administrator has to worry about it.

## **1.8 ABOUT PANDORA FMS**

---

*Pandora FMS original project was started by Sancho Lerena at 2003. At present other people are working on it: Esteban Sánchez, Ramón Novoa, Jorge González, Raúl Mateos, David Villanueva, Manuel Arostegui, José Ángel de Bustos, José Navarro, Jonathan Barajas and many others. We want to thank many other people who help us with translation, graphic design, bug reporting and interesting ideas. Please visit our website for full credits.*

Pandora FMS is Open Source, and is published under GPL License version 2. In order to know the last changes, visit [the official web site of the project in Sourceforge](#).



## 2 PANDORA FMS SERVER INSTALLATION

---

### 2.1 PREREQUISITES

---

Pandora FMS is not only a single app, it is made up by several shellscript files (UNIX/Linux Agents), a Web application in PHP (Console), some code in C++ (Windows Agent), some code in PERL5 (Server) and some structure and data in SQL (Database), so, to get all this running you need to have some pieces of software installed in your system. This is a list of packages, libraries and software you need before installing Pandora FMS.

#### 2.1.1 Pandora FMS Data Server

To work with Pandora FMS Data Server you need to have the following Perl modules installed in the system. This packages could be installed using your distribution packaging system or using CPAN.

- XML::Simple, useful XML functions.
- Digest::MD5, MD5 generation.
- Time::Local, basic Date and Time manipulation.
- DBI, DB interface with MySQL.
- Date::Manip, Date and Time handling and manipulation.
- threads and threads::shared, Pandora FMS has multithread coding and uses shared memory to communicate between process (using locks). Systems like Gentoo do not use threaded Perl distribution by default, but could be installed easily emerging it.

You can find all of them at [CPAN web page](#) or install them using your default package installation system. These packages are in the default distribution Suse 9.1 and Debian 3.0 GNU/Linux. Also available for Solaris in the CPAN repository. It is also probable that you need to set the TZ (Time Zone) environment variable.

#### 2.1.2 Pandora FMS Network Server

Requires SSH Server and Perl v5.8 or higher and the next Perl modules:

- IO::Socket, management and manipulation of TCP/UDP sockets.
- Time::HiRes, needed for ICMP times.
- Time::Local, basic Date and Time manipulation.
- SNMP, for SNMP management.
- Date::Manip, needed to manipulate Date and Time formats of input, output and comparison.
- Net::Ping, to calculate latency times (the Server has tu run as user root).
- threads and threads::shared.

The package net-snmp is also needed to use SNMP fuctions. It is worth to say that to run

GENERIC\_ICMP\_DATA module type (ICMP latency time calculation) Pandora FMS Network Server must run with root privileges.

### 2.1.3 SNMP trap reception Console

You need to install the `net-snmp` package, which is included in all GNU/Linux distributions. You have to use the `snmptrapd` binary provided in that package. Before installation, check the launcher script for the SNMP trap reception Console of Pandora FMS and be sure that the location of `snmptrapd` binary is correct on your distribution, being usually `/usr/sbin/snmptrapd`. This file location is defined in variable `DAEMON_PATH`.

This binary gets the SNMP traps, generating a log parsed by the Pandora FMS SNMP Console. It has the following Perl module dependencies:

- `Date::Manip`
- `Time::Local`
- `Time::HiRes`
- `threads` and `threads::shared`

### 2.1.4 Network Reconnaissance Server (*Recon*)

The Network Reconnaissance Server of Pandora FMS is a small piece in charge of discovering systems sweeping the network using ICMP. Therefore it needs some Perl dependencies:

- `Socket`, default on almost all Perl distributions.
- `Posix`, default on almost all Perl distributions.
- `threads` and `threads::shared`, default on almost all Perl distributions, except in some systems (Gentoo).
- `Net::Ping`, default on almost all Perl distributions.
- `Time::Local`, default on almost all Perl distributions.
- `NetAddr::IP`, to manage IP addresses, networks and network masks. Included in Pandora FMS packaged versions (RPM). In other systems, like Debian and Ubuntu, it is available as `.DEB` package. It can be installed from CPAN.

### 2.1.5 Installing dependencies

It is easier to install native packages (`.RPM` or `.DEB`) instead of CPAN, but if you know CPAN, it could be faster to use CPAN interface to satisfy dependencies. To use CPAN you need to launch `cpan`, answer to all setup questions, and use a command line to install each Perl module, it has a syntax like:

```
install NetAddr:IP
```

If you want to use CPAN to build Perl modules, please install BEFORE `make`, `autotools`, a C Compiler and other tools needed to build Perl modules from source code.

Whatever method you use, you **MUST** resolve dependencies installing missing packages. In Debian systems, all dependencies should be fixed with the following command:

```
apt-get install libdate-manip-perl snmp libsnmp-perl
libtime-format-perl libxml-simple-perl libnetaddr-ip-perl
```

Note: In Ubuntu 7.0x *Net::Ping* is installed by default.

## 2.2 AUTOMATIC INSTALLATION USING INSTALLER

---

Installing Pandora FMS with the installer, available in the SVN repository, or installing it with any of the available .rpm and/or .deb packages, produces the same result. The full and automatic installation of the system, where you only need to configure it to make it work.

The installer it is in the Pandora FMS servers directory, you can execute it without arguments to see how it works:

```
$ ./pandora_server_installer

Pandora FMS Server Installer (c) 2008 Manuel Arostegui
This program is licensed under GPL2 Terms. http://pandora.sourceforge.net

--install      To install Pandora FMS Servers on this system (You have to be root)
--uninstall    To uninstall and remove Pandora FMS Servers on this System
```

Run it as *root* and with *--install* parameter:

```
#!/pandora_server_installer --install
Pandora FMS Server Installer (c) 2008 Manuel Arostegui
This program is licensed under GPL2 Terms.
http://pandora.sourceforge.net

cp bin/pandora_recon blib/script/pandora_recon
/usr/bin/perl "-MExtUtils::MY" -e "MY->fixin(shift) "
blib/script/pandora_recon
cp bin/pandora_snmpconsole blib/script/pandora_snmpconsole
/usr/bin/perl "-MExtUtils::MY" -e "MY->fixin(shift) "
blib/script/pandora_snmpconsole
cp bin/pandora_server blib/script/pandora_server
/usr/bin/perl "-MExtUtils::MY" -e "MY->fixin(shift) "
blib/script/pandora_server
cp bin/pandora_network blib/script/pandora_network
/usr/bin/perl "-MExtUtils::MY" -e "MY->fixin(shift) "
blib/script/pandora_network
Installing /usr/local/bin/pandora_server
Installing /usr/local/bin/pandora_recon
Installing /usr/local/bin/pandora_snmpconsole
Installing /usr/local/bin/pandora_network
Writing /usr/local/lib/perl/5.8.8/auto/PandoraFMS/.packlist
Appending installation info
to /usr/local/lib/perl/5.8.8/perllocal.pod
```

## 2.3 MANUAL INSTALLATION OF PANDORA FMS SERVERS

---

Using the tarball and untaring it in */tmp/pandora\_server*, and placing it on that directory. It has a structure like:

- *./lib* - Internal Pandora FMS Perl libraries. Should be copied to default PERL library path, usually something like */usr/lib/perl5*. Using the Perl Makefile should be easy to install that libraries automatically.
- *./bin* - Pandora FMS executables (Perl code). Should be copied directly to */usr/bin*. Makefile should do this operation automatically.
- Launch scripts, called *pandora\_network*, *pandora\_server*, *pandora\_snmpconsole* and *pandora\_recon*. Should be copied/placed to */etc/init.d* and **this has to be done manually as described below**.
- *./util* - With some utilities, that should be used (specially *pandora\_db.pl*, that is the DB maintenance script). This has to be copied manually as described below in the directory */usr/share/pandora\_server*.
- *./conf* - Contains a default configuration file (*pandora\_server.conf*) that should be modified. This has to be copied manually as described below, in the directory */etc/pandora*.
- *Makefile.PL*, this is the Perl script that checks the dependencies and copies the libraries and binaries to their default system paths. It creates a standard *Makefile* that allows to launch the command *make* to do the final installation, as described in the following steps.

To start with the installation, you have to untar the tarball, that contains Pandora FMS servers, to the directory */tmp* for example. This directory can be deleted after installation.

### 2.3.1 To create user "pandora" and set up permissions

If you don't have a "pandora" user yet, create it with the command:

```
useradd pandora -d /home/pandora
```

And apply specific permissions to incoming directory and set ownership for this directory:

```
mkdir /home/pandora
mkdir /home/pandora/.ssh
chown -R pandora /home/pandora
```

You, also, have to create the directory */var/run/pandora/* and */var/log/pandora* and make sure that the permissions are correctly set for the first one:

```
mkdir /var/spool/pandora
mkdir /var/spool/pandora/data_in
mkdir /var/log/pandora
chown pandora /var/spool/pandora/data_in
```

```
chmod 700 /var/spool/pandora/data_in
```

SSH and Tentacle transfers to the server will use this user, **so it will need a strong password**. If you want to use FTP for transfers, be sure that "pandora" user can reach the incoming directory.

Add the public key of each Agent that will send data to Pandora FMS Server, in the file */home/pandora/.ssh/authorized\_keys*.

These keys must be SSH v2, OpenSSH DiffieHellman (DF) or RSA. To convert keys you can use the *ssh-keygen* tool. Pandora FMS server will check and parse the XML files sent by the agents, and will insert them into the Pandora FMS database.

A user without privileges can run Pandora FMS Data Server, you can use the user "pandora", it only needs to be able to run */usr/bin/perl* and access to */etc/pandora/pandora\_server.conf* file and */var/spool/pandora/data\_in* directory.

Otherwise, Pandora FMS server will complain about either the missing directory or the wrong permissions.

### 2.3.2 Installing executables and libraries on your system

It is necessary a complete "make" system. If you don have one, install it. If you cannot use or install one, please follow the directions in the section "Manual installation without makefile", to know how to copy the executables and libraries to your system.

#### *Using makefile*

In Pandora FMS ditribution directory, execute the command:

```
perl Makefile.PL
make
```

And as root:

```
make install
```

It should copy the executables to your default system path, and libraries to your default Perl system library path.

#### **Manual install without makefile**

Copy the *bin/\** directory to */usr/bin* (with root privileges):

```
cp bin/* /usr/bin
```

Copy Pandora FMS Perl libraries directory from *lib/PandoraFMS* to your Perl library path, usually */usr/lib/perl5/*:

```
cp -R lib/PandoraFMS /usr/share/perl5/
```

### 2.3.3 Setting up *snmptrapd* for Pandora FMS SNMP Console

Pandora FMS SNMP Console uses *snmptrapd* to grab SNMP traps. *snmptrapd* is a standard tool, present on almost all UNIX systems, to grab traps and write a logfile. Pandora FMS configures *snmptrapd* to write a custom logfile and reads it every x seconds, executing alerts if defined.

Previously, *snmptrapd* will accept all incoming notifications, and log them automatically (even if no explicit configuration is provided). Starting with 5.3 release, access control checks will be applied to incoming notifications.

If *snmptrapd* is run without a suitable configuration file (or equivalent access control settings), then **such traps will not be processed**.

Probably you need will to configure your *snmptrapd* using the file */etc/snmp/snmptrapd.conf*. If doesn't exist, check */var/log/pandora/pandora\_snmp.log* file for warnings or errors.

A basic *snmptrapd.conf* could be like:

```
authCommunity log public
```

### 2.3.4 Testing Pandora FMS Server installation

If the installation is complete, it is a good idea to perform small operations to check that everything is correct. To do so, start one of the servers, which should show a similar message as the following (can be a bit different because of versions):

```
# pandora_server --help

Pandora FMS Data Server 1.3-dev Build PS070731 Copyright (c) 2004-2007 ArticaST
This program is Free Software, licensed under the terms of GPL License v2.
You can download latest versions and documentation at http://pandora.sourceforge.net.

I Need at least one parameter: Complete path to Pandora FMS Config file
```

Congratulations, Pandora FMS Server is ready to work.

### 2.3.5 Testing Pandora FMS trap reception Console

After setting up the SNMP trap reception Console, it is a good idea to try to send some traps form the localhost where both, the console and the trap reception daemon, are installed, and then send from a remote system some traps to check the connectivity and the correct configuration of the system trap reception daemon. To get additional information about how to work with SNMP under GNU/Linux, check the additional documentation [\[1\]](#).

If we have our trap console at 192.168.5.2, we can send some traps with different values, as follows:

```
snmptrap -v 1 -c public 192.168.5.2 1.3.6.1.2.1.2.2.1.1 192.168.5.2 6
666 1233433
snmptrap -v 1 -c public 192.168.5.2 1.3.6.1.2.1.2.2.1.1 192.168.5.2 6
```

```
666 1233433 .1.3.6.1.2.1.2.2.1.1.6 i 234234234
snmptrap -v 1 -c public 192.168.5.2 .1.3.6.1.4.1.2789.2005
192.168.5.2 6 666 1233433 .1.3.6.1.4.1.2789.2005.1 s "Any text"
```

We can try independently of the trap reception console of Pandora FMS with the system daemon *snmptrapd* and generate a log compatible with Pandora FMS, and then see manually the result of that file:

```
snmptrapd -f -t -On -n -a -Lf logfile.txt -p pidfile -F "%02.2m/
%02.2l/%y %02.2h:%02.2j:%02.2k %B %N %w %W %q %v \n"
```

This generated log would have this format:

```
08/18/2005 03:01:14 192.168.5.2 .1.3.6.1.2.1.2.2.1.1 6 Enterprise
Specific .666 .1.3.6.1.2.1.2.2.1.1.6 = INTEGER: 234234234
08/18/2005 03:02:37 192.168.5.2 .1.3.6.1.4.1.2789.2005 6 Enterprise
Specific .666 .1.3.6.1.4.1.2789.2005.1 = STRING: "Any text"
```

## 2.4 CONFIGURING PANDORA FMS SERVER

---

After installing Pandora FMS Server, you will need to edit the file *pandora\_server.conf*, where the variables of the server configuration are defined. The file *pandora\_server.conf* is a plain text file, which you can edit with your preferred text editor, like *vi*, *emacs* or *kate*. This configuration file is common to all Pandora FMS Servers (Data server, SNMP Server, Network server and others in the future), you can also have different copies of the configuration file for each Pandora FMS Server you have (in an advanced setup).

Create the directory */etc/pandora*:

```
mkdir /etc/pandora
```

Copy the default config file (*pandora\_server.conf*) to */etc/pandora* directory:

```
cp conf/pandora_server.conf /etc/pandora/
```

Edit Pandora FMS Server configuration file to */etc/pandora/pandora\_server.conf* and take a look at the lines:

- **dbuser**: with database user (default to pandora).
- **dbpass**: with database password (change default!).
- **dbhost**: Database hostname or IP.
- **dbname**: Database name (default to pandora).

Please, note that the Database must be created before starting the server, otherwise, the server will not start properly and it will be stopped. To find out how to create the database please go to [Installing Pandora FMS Database](#).

For security reasons it is not recommended to use the default values (especially the password). It is very important to check that there are no blank spaces before the server configuration words and before the data, for this configuration item.

Other items for the configuration file are:

- **incomingdir**: place where the XML data files arrive. The owner must be the user "pandora", therefore it is a good idea to run Pandora FMS Data Server with this user. If a SSH server is used to receive the data, then the remote user must be "pandora" as well. If Tentacle is used to receive the data, then also the Tentacle user must be "pandora".
- **logfile**: Server logfile.
- **snmp\_logfile**: *snmptrapd* logfile.
- **errorlog\_file**: Fatal error logfile, where the error messages are stored.
- **master**: This variable sets the Master Mode in the Network Server. This allows the Master Server to take over the tasks of servers not online, allowing Pandora FMS to operate in HA mode.
- **checksum**: Activate or deactivate checksum mode (feature from 1.0 version that currently has no use). Deprecated, probably will disappear in future versions.
- **snmpconsole**: Enables *snmpconsole* for this configuration file.
- **networkserver**: Enables the Network Server for this configuration file.
- **reconserver**: Enables the Recon Server for this configuration file.
- **dataserver**: Enables the Data Server for this configuration file.
- **network\_timeout**: Timeout (in seconds) for network tasks, such as pings, snmp or tcp checks. If you set it too high, it can cause bottlenecks on heavy-loaded systems. A good value for LAN is 3 seconds.
- **server\_keeplive**: Timeout (in seconds) to consider a non-responding Pandora FMS server and take it as down (used to boost HA and to render down servers in console).
- **server\_threshold**: Internal sleep loop for servers. It cannot have a higher value than *keeplive* because servers will seem to be down, set a low or zero value for heavy loaded systems.
- **network\_threads**: Internal threads of Network Server to execute tasks. Do not set it too high (>20) because it could have a big performance penalty. A good initial value is 5, for almost every system. This generates heavy CPU load on the Server for low thresholds and many threads.



### 2.4.1 New options introduced in 1.3.1 version

- **icmp\_checks:** It defines the number of "pings" that each `icmp_proc` module performs. If you specify a number higher than one, it means that in case that the first one fails, it performs the next try. If the specified number is "5" and the first one fails, but not the second one, only two pings are performed. To consider the operation as "fail", all the specified pings must fail. If the number specified is very high, then bottlenecks in the Server can occur and there can be a lot of traffic on the network. The default value is 2.
- **alert\_recovery:** 1 or 0. Defines if Pandora FMS launches the other alert when the alert condition is recovered previously. It has the same fields, but adds the string "[RECOVER]" to field2 and field3. Is disabled by default, and activated with a value "1".
- **tcp\_checks:** Number of TCP retries if first attempt fails.
- **tcp\_timeout:** Specific timeout (in seconds) for TCP requests.
- **snmp\_timeout:** Specific timeout (in seconds) for SNMP request.
- **snmp\_checks:** Number of SNMP retries if first attempt fails.
- **snmp\_proc\_deadresponse:** 1 by default. It returns *DOWN* if it cannot connect, or if *NULL* is returned after a *SNMP\_PROC* query (usually to check the interface status, etc).

All this arguments, especially the last ones, that allow to adjust the behaviour of the network checks, allow when the installation has different Network Servers, in different areas of the network set different timeouts and retries to fit different requirements in big companies: specific network servers for WAN environments with high latency times and high packet losses, network servers for LANs with small latency times and without packet losses, servers that work with slow SNMP devices, etc.

**Remember:** you have to create the directory `/var/spool/pandora/data_in` where Pandora FMS Server will read and write data, sent by remote agents using Tentacle, SSH or FTP. This directory must be owned by the user "pandora", or the user must be allowed to write on it.

Pandora FMS servers do not need to run as *root*, except the Network Server (see below) and the Pandora FMS SNMP Console, because they need to open UDP port 161 (this can be solved setting `SUID0` to the `snmptrapd` binary) and running the rest of the Server using a user without privileges or launch `snmptrapd` and *Pandora FMS SNMP Console* in separate scripts.

Also Pandora FMS Network Server can be run using a user without privileges, but the ICMP data types won't work, since root privileges are required to create ICMP packets.

Check the MySQL connection with the correct user and password before running the server, using the following `mysql` command (database password will be asked):

- `mysql -u pandora -p -D pandora`

Pandora FMS Data Server distribution tarball includes a Posix/System V start/stop script to "daemonize" Pandora FMS Server. It is possible that you need to customize it, but it has been tested, and runs fine on several GNU/Linux (Debian, Suse, Redhat, Ubuntu) and UNIX systems, like Solaris.

It has start|stop|restart parameters to include it in your default init level directory, and it creates a logfile defined in the variable *LOGFILE* (*/var/log/pandora/pandora\_server.log* by default).

The log directory must be created:

```
mkdir /var/log/pandora/
```

Copy startup daemons to */etc/init.d*:

```
cp pandora_* /etc/init.d/
```

Create links to appropriate *runlevel* to be sure that Pandora FMS starts when the system boots up (i.e., level 2 on a Debian-like system):

```
ln -s /etc/init.d/pandora_server /etc/rc2.d/S90pandora_server
ln -s /etc/init.d/pandora_recon /etc/rc2.d/S90pandora_recon
ln -s /etc/init.d/pandora_network /etc/rc2.d/S90pandora_network
ln
-s /etc/init.d/pandora_snmpconsole /etc/rc2.d/S90pandora_snmpconsole
```

Note: In some systems it is a good idea to setup a link in levels 6 and 0 to kill services and create a */var/run/pandora* directory in each startup. It depends on system kind.

Create shared resources directory for Pandora FMS:

```
mkdir /usr/share/pandora
```

Copy util directory to */usr/share/pandora*

```
cp -R util /usr/share/pandora
```

## 2.4.2 Configuring Tentacle to receive data packages

Pandora FMS 1.3.1 introduces the use of the **Tentacle** project to receive the data packages sent by the agent. In older Pandora FMS versions, the default transfer method was SSH. Now Tentacle is used due to its simple use.

Tentacle is a client/server file transfer protocol that aims to be:

- Secure by design.
- Easy to use.
- Versatile and cross-platform.

Tentacle was created to replace more complex tools like SCP and FTP for simple file transfer/retrieval, and switch from authentication mechanisms like .netrc, interactive logins and SSH keys to X.509 certificates. Simple password authentication over a SSL secured connection is supported too.

The client and server are designed to be run from the command line or called from a shell script, and no configuration files are needed.

Tentacle is now the default file transfer method for [Pandora FMS](#) and [Babel Enterprise](#).

Tentacle is implemented in Perl and ANSI C (Windows platforms included).

You can download it and get more information at the official Sourceforge website [\[2\]](#).

### 2.4.3 Configuring Tentacle Server to use it with Pandora FMS Data Server

The instructions to install Tentacle Server are in the [Tentacle User Guide](#).

Once the server is installed, the script *tentacle\_serverd*, located at */usr/share/pandora/util* can be used to start it automatically every time the system boots up.

Before copying the script is a good idea to edit it, and check that the values of the variables *PANDORA\_SERVER\_PATH*, *TENTACLE\_DAEMON*, *TENTACLE\_PATH*, *TENTACLE\_USER*, *TENTACLE\_ADDR* and *TENTACLE\_PORT*, located at the beginning of the script, are correct.

Extra parameters can be added for advanced configurations, to do so, use the Tentacle server variable *TENTACLE\_EXT\_OPTS*.

To install the script run:

```
cp /usr/share/pandora/util/tentacle_serverd /etc/init.d/  
ln -s /etc/init.d/tentacle_serverd /etc/rc2.d/S90tentacle_serverd
```

### 2.4.4 Setting up SSH configuration to receive data packets

Pandora FMS can use the SSH protocol to copy XML data packets, generated by the agents, to the server. You need to generate a SSH2 key in every agent, and copy the public key in */home/pandora/.ssh/authorized\_keys*, so you need to create a user called "pandora" without privileges. The agents will use this user to copy data into Pandora FMS Data Server directory */var/spool/pandora/data\_in*.

You must BE SURE that user "pandora" exists (if not, create it using the command *useradd*), that */home/pandora/.ssh/authorized\_keys* exists, and the owner of this file and directory is "pandora", finally set permissions for both file and directory to 600.

Please also be sure that directory */var/spool/pandora/data\_in* exists and "pandora" user can write in it.

## 3 INSTALLING PANDORA FMS CONSOLE

---

### 3.1 INSTALLING PANDORA FMS DATABASE

---

In the new 1.3 version of the Pandora FMS Web Console the process of creating and setting up the database can be easily made from a browser, following the same Wizard that installs and sets up the console. This process is described in the console installation section below. We recommend you to use the Wizard instead of doing it manually.

Anyway, this is the process to create all the database stuff from the command line.

Please look at [MySQL install and management guide](#) to obtain information about how to create a MySQL database, how to manage MySQL users and give them privileges to read/write in Pandora FMS Database. Remember that you must write the password of the root user in MySQL database to enter mysql command line. This user is not the same of the Operating System. By default, the root password of MySQL is empty (for almost every distribution), you must change this password with the MySQL command *mysqladmin*. Please be careful with this.

You need a database named "pandora", you can rename it, although you would have to reconfigure the server too.

To create Pandora FMS database structure in MySQL Server you have the SQL script "pandoradb.sql".

It creates tables and indexes needed to insert information into Pandora FMS database.

You must populate the database with the SQL script "pandoradb\_data.sql", it inserts data needed to run the Web Console and a default user (login: **admin**, password: **pandora**) to access Pandora FMS Web Console.

First, create a database called "pandora", and set a user able to access this database:

```
mysql> create database pandora;
```

Then, execute the next commands using a user with enough privileges to create tables and indexes for pandora Database into your MySQL Server:

```
cat pandoradb.sql | mysql -D pandora -u root -p
cat pandoradb_data.sql | mysql -D pandora -u root -p
```

Then, execute the next commands using a user with enough privileges to create tables and indexes for pandora Database into your MySQL Server:

```
mysql> source path_to_pandora_dbstruct.sql
mysql> source path_to_pandora_dbdata.sql
```

This example is valid using root user in MySQL. Now you have to create the user "pandora" and give him privileges from the localhost:

```
mysql> grant all on pandora.* to 'pandora'@'localhost' identified by 'pandora';
```

Keep in mind that users need access to Pandora FMS Web Console and to Pandora FMS Server. If your deployment has many subcomponents in different physical machines, you have to setup a MySQL user with privileges to access from different locations.

If you get the error:

```
Warning: mysql_connect() [function.mysql-connect]: client does not support authentication protocol requested by server; consider upgrading
```

when authenticating to the Web Console, you have to change the way the password is stored into the database:

```
mysql> set password for 'pandora'@'localhost' = old_password('pandora');
```

Please note this user will be used by several Pandora FMS subcomponents (Pandora FMS Server and Pandora FMS Web Console) to access the database.

## 3.2 PANDORA FMS WEB CONSOLE INSTALLATION

---

Before installing Pandora FMS Web Console, you need the following dependencies and software:

- Web server. Apache2 is recommended.
- PHP 4.3.x, or PHP 5.x. Both have been tested with Pandora FMS, but 5.x is recommended.
- PHP modules for MySQL, GD, session management and SNMP.
- PEAR PHP library.

To install Pandora FMS Web Console, simply untar the file in your HTTP server publishing directory and set permissions to "www-data" or "http user".

Then open your browser at:

```
http://host:port/installdir/install.php
```

Wizard installation sets up your console in five steps. If everything goes right, you should be able to access to your Pandora FMS Web console at:

```
http://host:port/installdir/index.php
```

The first time you log in there is a default admin user "admin" and its password is "pandora". It is worth to say that **YOU MUST CHANGE CREDENTIALS BEFORE LOGIN THE FIRST TIME**, change them or create another account, give it administrator privileges, and disable or delete this one.



If you cannot see a screen like this, it is possible that you have problems with your PHP installation. Please check that the PHP engine is running after installing the web. First try to access to the server IP with a browser. You must see the Apache Welcome page or some kind of valid response.

Remember that after installing PHP and the PHP module for Apache you must stop and start Apache server. As an example under Ubuntu with Apache2:

```
/etc/init.d/apache2 stop  
/etc/init.d/apache2 start
```

To check PHP and Apache integration you can create the file test.php with the following lines:

```
<?PHP  
    echo "<h1>TEST</h1>";  
    phpinfo();  
?>
```

Now, copy this file to the Apache HTTPDOC directory. This directory depends of the Operating System or GNU/Linux Distribution, for example in Ubuntu and Debian this directory is `/var/www`, and in SUSE is `/srv/www/htdocs`. Red Hat based distributions use `/var/www/html`.

To check this integration, please use your browser to open the following URL:

```
http://IP/test.php
```

Where IP is the IP address of your Apache server. If the integration is correct, you will see the following text string in the browser: "TEST"; and a big table with a lot of info about your PHP installation.

## 4 INSTALLING AND USING PANDORA FMS

### PHYSICAL AGENTS

---

#### 4.1 UNDERSTANDING WHAT IS A PANDORA FMS AGENT

---

Pandora FMS agents collect all the data from the systems. They are executed in each local system, although they can also collect remote information, installing monitoring systems for the agent in several different machines called satellite agents.

They are developed to work under a given platform, making use of the specific tools of the language used: VBScript/Windows Scripting for Microsoft platforms (Win2000, WinXP and Win2003), ShellScripting for UNIX includes GNU/Linux, Solaris, AIX, HP-UX and BSD, as well as Nokia's IPSO. Pandora FMS agents can be developed in any language, given its simple API system and being open source. There are branches of the Pandora FMS project started to create agents in Posix C, Perl and Java for those systems that require closed agents.

**Pandora FMS Agents are Free Software**, i.e., the way the agents collect and send the information is documented. An agent can be recreated in any programming language, and can be upgraded easily, to improve aspects of the program not covered so far.

This document describes the installation of agents in machines running under Windows and UNIX operating systems.

##### 4.1.1 Generic role of the agents

Regardless of the platform where an agent is running on, this is formed up by the following elements:

- A script (or a binary application in Windows) that collects and sends the data to the server. For UNIX machines, the script is called *pandora\_agent.sh* and is executed directly from the Pandora FMS agent folder.
- One or several configuration files where the values to be collected are defined. The file is called *pandora\_agent.conf* both for Windows and UNIX machines.

This simple structure makes easy the customization of an agent. There is no need to code again the agent to modify the way it works, as the configuration file holds most of the parameters needed to do so.

## 4.2 PANDORA FMS AGENT CONFIGURATION

---

### 4.2.1 Main program

The main program is the executable file that collects the specified data in the configuration file. It sends the data to the server in XML. In Windows machines, the application is installed as a service and it is executed at the time intervals set in the configuration file. In machines running UNIX, the main program is a script that runs through a special script called *pandora\_agent*, and runs continuously in the machine as a process.

### 4.2.2 Configuration file

The data collection in the host system is the gathering of independent data units, defined in the */etc/pandora/pandora\_agent.conf* file. This file is divided in two parts:

- **General parameters:** Configure general options about the server location, the agent name, interval, and other general options.
- **Module definitions:** Configure and define the method of extraction for each piece of information that will be extracted from the local host and sent to Pandora FMS Server.

### 4.2.3 General parameters

The general parameters of the agent configuration are defined in this section. Some of these parameters are common for every system, and others are specific for Windows or UNIX. The general parameters are:

- **server\_ip:** The server IP is the IP address, or the host name, of the Pandora FMS Server, where the data will be stored. The host must be reachable and must be listening on port 22 (SSH) or 41121 (Tentacle).
- **server\_path:** The server path is the full path of the folder where the server stores the data sent by the agent. It is usually */var/spool/pandora/data\_in*.
- **temporal:** This is the full path of the folder where the agent stores the data locally, before it is sent to the server. It must be said that the data packages are deleted once the agent tries to contact the Pandora FMS Server, no matter if the communication was successful or not. This is done to avoid over flooding hard drive of the host system where the agent runs. The location of the local folder varies with the architecture of the host system. In Unix systems this is usually */var/spool/pandora/data\_out*, and in Windows systems *C:\program files\pandora\data\_out*.
- **interval:** This is the time interval, in seconds, that will use the agent to gather the data from the host and send it to the server. The recommended range is between 300 (5 minutes) up to 600 (10 minutes). This number can be higher, although it is



important to consider the impact of a big number in the database.

- **debug:** This parameter is used to test the generation of data files, forcing the agent not to copy the data file to the server, so you can check the data file contents and copy the XML data file manually. It does not delete any data when the process is finished, so the data file will be in the temporary directory. The activity is written in a log file. The file is named *pandora\_agent.log* (see *logfile* above).
- **checksum:** This parameter can take two values. If the value is 1, the checksums are performed through MD5. If the value is 0, the checksum is not performed at all. This may be useful for systems where a MD5 tool cannot be implemented. If the checksum is deactivated in the agent, it must be also disconnected in the server. Otherwise it could create problems.
- **agent\_name:** This is an alternative host name. This parameter is optional as it was not declared but obtained directly from the system. The parameter can be used to overwrite the host name in case there is any conflict.
- **encoding:** Sets the encoding type of your local system, i.e. iso-8859-15, or utf-8. This is only available for UNIX agents.
- **server\_port:** This parameter lets you specify the remote port of the server listening. It was 22 by default and for SSH, but right now is 41121 for Tentacle. In case you are not using Tentacle, or that the server is listening on other port, this is where you should change it.
- **transfer\_mode:** This parameter lets you specify the transfer mode installed to send the data from the agent to the server. Available modes are: **ssh** (using SCP), **tentacle**, **ftp**, or **local**. Local mode is only for systems where the agent runs in the same machine as the server does, because is basically a copy between directories. Local mode is available only for GNU/Linux agents.
- **server\_pwd:** Specific for Windows FTP and for Tentacle transfer mode, although the password is optional for the last one. Server password for password authentication.
- **server\_ssl:** Specific for Tentacle transfer mode. It allows to enable (1) or disable (0) the encryption over SSL connections.
- **server\_opts:** Specific for Tentacle transfer mode. It allows to use extra parameters with Tentacle client, for advanced configurations. They must be quoted (i.e. *"-v -r 5"*).
- **delayed\_startup:** This parameter lets you configure the Pandora FMS Agent to start running after an amount of time (in minutes) after you run it manually. This could be useful for systems with a high load. By default it is disabled, that is, Pandora FMS Agent will start running at the very moment you run it manually. This is only available for UNIX agents.
- **pandora\_nice:** This parameter lets you specify the priority the Pandora FMS Agent process will have in your system. This is only available for UNIX agents.
- **ftp\_password:** Specific password for FTP transfer mode. This is only available for

Windows agents. **NOTE: this parameter has dissapeared in Pandora FMS 1.3.1, deprecated thanks to *server\_pwd*.**

An example of the general parameters from a UNIX configuration would be:

```
server_ip      192.168.12.12
server_path    /var/spool/pandora/data_in
temporal       /var/spool/pandora/data_out
interval       300
agent_name     dakotaSR01
debug          0
checksum       0
```

#### 4.2.4 Module definition

Each data item to be collected must be defined precisely in each module, using the exact syntax. As many values as necessary can be set to be collected, adding at the end of the general parameters as many modules as the number of values to collect. Each module is made of several directives. The list below is a descriptive relation of all module marks available for UNIX agents (almost all of them are also applicable to the Windows agent).

##### **module\_begin**

Defines the beginning of the module.

##### **module\_name <name>**

Name of the module. This is the ID for this module, choose a name without blank spaces and not very long. There is no practical limitation (250 char max) but short names will be easier to manage. **This name cannot be duplicated** with a similar name in the same agent. This name could be duplicated with other modules in other agents.

##### **module\_type <type>**

Data type that the module will handle. There are four data types for agents:

- Numeric (*generic\_data*). Simple numeric data, float or integer. If the values are of the float type, they will be truncated to their integer value.
- Incremental (*generic\_data\_inc*). Integer numeric data equal to the differential between the actual value and the previous one. When this differential is negative, the value is set to 0.
- Alphanumeric (*generic\_data\_string*). Text strings up to 255 characters.
- Monitors (*generic\_proc*). Stores numerically the status of the processes. This data type is called monitor because it assigns 0 to an "Incorrect" status and any value above 0 to any "Correct" status.

**module\_exec <command>**

This is the generic "*command to execute*" directive. For both, UNIX and Windows agents, there is only one directive to obtain data in a generic way, executing a single command (you can use pipes to redirect the execution to another command). This directive executes a command and stores the returned value. This method is also available for Windows agents. This is the general purpose method for both kind of agents.

For a Windows agent there are more directives to obtain data, described below.

**module\_service <service>**

(Win32 Only)

Checks if a given service name is running in this host. Remember to use " " characters if service name contains blank spaces.

**module\_proc <process>**

(Win32 Only)

Checks if a given processname is running in this host. If the process name contains blank spaces do not use " ". Notice also that the process name must have .exe extension. The module will return the number of process running with this name.

**module\_freedisk <drive\_letter:>**

(Win32 Only)

Checks free disk on drive letter (do not forget ":" after drive letter).

**module\_cpuusage <cpu id>**

(Win32 Only)

Returns CPU usage on CPU number cpu. If you only have one CPU, use 0.

**module\_freememory**

(Win32 Only)

Returns free memory in the whole system.

**module\_min <value>**

This is the minimum valid value for the data generated in this module. If the module has not yet been defined in the web console, this value will be taken from this directive. This directive is not compulsory. This value does not override the value defined in the agent. If the module does not exist in the management console, it is created automatically when working on learning mode.

**module\_max <value>**

It is the maximum valid value for the data generated in this module. If the module has not been defined in the web console, this value will be taken from this directive. This directive is not compulsory and is not supported by the Windows agent. This value does not override the value defined in the agent. If the module does not exist in the management console, it is created automatically when working on learning mode.

**module\_description <text>**

This directive is used to add a comment to the module. This directive is not compulsory. This value does not override the value defined in the agent. If the module does not exist in the management console, it is created automatically when working on learning mode.

**module\_interval <factor>**

Since Pandora 1.2 introduced this new feature, for each module, you can setup its own interval. This interval is calculated as a multiple factor for agent interval. For example, if your agent has a 300 (5 minutes) interval, and you want a module to be processed each 15 minutes, you should add this line: *module\_interval 3*. So this module will be calculated each  $300\text{sec} \times 3 = 900\text{sec}$  (15 minutes).

**module\_end**

Defines the end of the module.

## 4.2.5 Examples

An example of a Windows module, checking whether the *EventLog* service is alive, would be:

```
module_begin
module_name ServicioReg
module_type generic_proc
module_service Eventlog
module_description Eventlog service availability
module_end
```

An example of a UNIX module would be:

```
module_begin
module_name cpu_user
module_type generic_data
module_exec vmstat | tail -1 | awk '{ print $14 }'
module_min 0
module_max 100
module_description User CPU
module_end
```

## 4.2.6 Types of Agents

With Pandora FMS it is possible to monitor virtually any system. This can be done either with a local agent collecting data directly from the system to be monitored, using a satellite agent collecting data from a system by SNMP, or using the new Pandora 1.2 agents (and newer), the remote agents, which can check services using remote network polling (TCP, UCP, ICMP/PING and SNMP), from the Pandora FMS Network Server.

The local agents can be either Windows or UNIX agents. The satellite agents can be implemented using any of the agents above. The modules are configured to collect data from xternal system by, for example, an SNMPGET tool.

## 4.3 INTRODUCTION TO UNIX AGENTS

---

The builtin UNIX applications and tools make the agents running on this system be very simple. There are also agents developed for AIX, GNU/Linux, Solaris and BSD platforms, some of them very similar but not identical.

In the list below there are a few comments for differences between UNIX versions.

### HP-UX

Has been tested without MD5.

### IBM AIX

MD5 signatures are used to guarantee the integrity of the generated data packages (this is optional). The MD5 package is integrated in AIX 5.1 and newer. There is a freeware package for AIX 4.3, but it has several issues and might not work correctly. In the case of having problems with the AIX agents the checksum system used to validate the integrity of the data can be disabled.

### SUN Solaris y OpenSolaris

The MD5 package is necessary to execute the Solaris agent correctly. This package is available from [sunfreeware.com](http://sunfreeware.com). It can also be downloaded for Solaris 8 from [following URL](#).

MD5 package installation on Solaris:

```
root@stest:/tmp:> gzip -d md5-6142000-sol8-sparc-local.gz
root@stest:/tmp:> pkgadd -d ./md5-6142000-sol8-sparc-local
```

```
The following packages are available:
```

```
1  SMCmd5      md5
   (sparc) 6142000
```

```
Select package(s) you wish to process (or 'all' to process all
```

packages). (default: all) [?,??,q]: 1

## Solaris SSH

The suggested SSH client is OpenSSH. If any other SSH client is to be used, it should be considered that each piece software may have different ways to generate and/or manage keys. For example, if F-Secure SSH is used, the public key must be in OpenSSH format when the keys are created. The format can be changed, from IETF to OpenSSH with F-Secure SSH, using the following command:

```
ssh-keygen -i -f file_ietf_pubkey
```

## GNU/Linux

SSH and MD5 should be installed in GNU/Linux by default, but if they are not, they can be installed using the tools available in each distribution.

## BSD (y Nokia IPSO)

SSH and MD5 should be installed by default. If they are not, it is necessary to install them.

## 4.4 INSTALLING PANDORA FMS UNIX AGENT

---

The software comes in a *.tar.gz* file. First of all, the file must be extracted into a folder, usually */tmp*. Then, once the file has been extracted, just run the Pandora FMS Agent Installer, called: *pandora\_agent\_installer*. To install it do:

```
./pandora_agent_installer --install
```

There is hardly any difference between AIX, Solaris and GNU/Linux, and they all work around the MD5 hash generation binaries.

After running the installer the main directory is */usr/share/pandora\_agent/* where Pandora FMS Agent is installed. The other folders are:

- */var/spool/pandora/data\_out*: folder where the data collected by the agents is stored.
- */usr/share/pandora\_agent/doc*: folder with information about the agent and its license.
- */etc/pandora/pandora\_agent.conf*: file where the data to be collected is defined, along with the command to be executed for the data collection. This is the system core, as it defines the main data to be collected in any Firewall.
- */etc/pandora/pandora\_user.conf*: file where several of the parameters to collect data from the monitored system are defined in detail.
- */usr/share/pandora\_agent/pandora\_agent*: this is the actual Pandora FMS agent. This file is a shellscript that collects the data configured in *pandora\_agent.conf* and

*pandora\_user.conf* files. It also transfers the data packages to the Pandora FMS Server, it's linked to */usr/bin/pandora\_agent*.

- */etc/init.d/pandora\_agent\_daemon*: start/stop script. It calls *pandora\_agent*. It offers two options, *start* and *stop*. In AIX systems the daemon is */etc/rc.pandora\_agent\_daemon*.
- */var/log/pandora/pandora\_agent.log*: text file where the activity of the Pandora FMS agent is saved, when the agent is executed in debugging mode.

#### 4.4.1 First running of the UNIX agent

To start the agent it is only necessary to execute :

```
etc/init.d/pandora_agent_daemon start
```

Pandora FMS Agent creates a file ("*/var/run/pandora/pandora\_agent.pid*") with the PID number of the process when it is started.

For IPSO systems the agent will be started with a nice -10 priority, so it becomes the process with the lowest priority over the system CPU. It will be executed when no other processes with a higher priority are waiting in the system CPU queue. IPSO agent has special parameter (*harmless\_mode*) to special management of CPU process on Checkpoint/NOKIA Systems. This is an very special case.

In BSD systems the maximum priority is +20 and the lowest -20.

To stop the agent, execute "*pandora\_agent\_daemon stop*" from "*/etc/init.d/*".

```
ser@server:~# /etc/init.d/pandora_agent_daemon stop
```

#### 4.4.2 Advanced configuration for UNIX Agent

The real power of Pandora resides in the capability of the agents to run user defined scripts. This could be used to collect specific data or to perform an operation to return any desired value. This is the purpose of *pandora\_user.conf*.

This file is executed every time in agent loop. It is a shell-script in which any command can be executed, as long as the output is in the XML format the agent uses to send data to the server. The XML structure would be:

```
<module>
<name>NAME</name>
<type>TYPE</type>
<data>DATA</data>
</module>
```

Where *NAME*, *TYPE* and *DATA* are the variables already defined in previous sections. The XML must be built manually, usually using *echo* commands.

For example, this would be the script a customized agent would use for Checkpoint FW1 in IPSO agents:

```
#!/bin/sh
# Pandora User-Defined acquisition script
# This code is under GPL licence
# Please refer documentation for more example and a more
# depth usage instructions

# mbuf clusters usados (%)
MBUF_TOTAL=`netstat -m |grep "mbuf cluster" | tr -s "/" " " |awk '{ print $2
}'`
MBUF_USED=`netstat -m |grep "mbuf cluster" | tr -s "/" " " |awk '{ print
$1 }'`
MBUF_USED_PER=`echo $MBUF_TOTAL $MBUF_USED | awk '{ print $2 / ($1 / 100) }

echo "<module>"
echo "<name>MBUF_CLUSTER_USED_PER</name>"
echo "<data>$MBUF_USED_PER</data>"
echo "<type>generic_data</type>"
echo "</module>"
```

A more complex example could be:

```
#!/bin/sh
# Pandora User-Defined acquisition script
# This code is under GPL licence
# Please refer documentation for more example and a more
# depth usage instructions

# Calculating the number of packages generated by ETH2,
# if nothing is generated
# within 20 seconds an alert is rosen
# Perform the calculation between 8 to 23h. Return ok for times
# outside this range

echo "<module>"
echo "<name>Packet_Generator_Check</name>"
echo "<type>generic_proc</type>"
UNO=`ifconfig eth2 | grep "TX packets" | cut -f 2 -d ":" | grep -o -e
"[0-9]*"`
sleep 20
DOS=`ifconfig eth2 | grep "TX packets" | cut -f 2 -d ":" | grep -o -e
"[0-9]*"`
HORA=`date +%k`
if [ "$HORA" -lt "8" ] && [ "$HORA" -gt "11" ]
then
    # Time out of range, no checking, everything OK
    # Fuera de hora, no compruebo, esta OK
    echo "<data>1</data>"
else
    if [ "$UNO" == "$DOS" ]
    then
        echo "<data>0</data>"
    else
        echo "<data>1</data>"
    fi
fi
echo "</module>"
```



## Implementation examples for UNIX Agents

Example #1: calculate the number of HITS of the main page of an Apache Web server (could degrade performance of huge logs).

```
module_begin
module_name WEB_Hits
module_type generic_data_inc
module_exec cat /var/log/apache/access.log | grep "index" | wc -l
module_end
```

Example #2: check if the process of the DNS server (named) is active or fell over:

```
module_begin
module_name DNS_Daemon
module_type generic_proc
module_exec ps -Af | grep named | grep -v "grep" | wc -l
module_end
```

## 4.5 PANDORA FMS WINDOWS AGENTS

---

### 4.5.1 Build Pandora FMS Windows Agent from sources

In order to build from sources, you will need the latest **Dev-Cpp IDE** version, along with **MinGW** tools. Download them from the [website](#).

Open the file *PandoraService.dev* with Dev-Cpp and build the project. Everything should compile fine in a default installation.

### 4.5.2 Installing Pandora FMS Windows Agent (*installer*)

From Pandora FMS v1.2.0, Windows version comes with an automatic installer, provided with the excellent Install Jammer free software, therefore the installation is very easy. You only need to choose a destination path, install and generate manually SSH keys as described below. For personalized or corporate deployments, you can also create your own installer (we provide Jammer sources for creating your own installable, so you can include a set of SSH keys in your own installer package).

### 4.5.3 Manual Pandora FMS Windows Agent installation

Before running or installing Pandora FMS Windows service, you must create the configuration directory and extract *PandoraBin.zip* file into it. No matter where is installed, because Pandora FMS Agent will adapt to any local directory. In the examples, the application will be installed in *C:\Pandora\*.

This directory will hold the configuration files, which are:

```
c:\Pandora\pandora_agent.conf      :: Pandoramain configuration
c:\Pandora\id_dsa                  :: Private SSH key
c:\Pandora\id_dsa.pub              :: Public SSH key
```

To install manually (without the installer) the Pandora FMS Windows Agent execute this sentence in a Windows command line:

```
PandoraService.exe --install
```

The Agent will be installed into the Windows services system. You can check it on Control Panel -> Administrative tools -> Services.

To run the Agent, open the "Services" dialog: (Control Panel -> Administrative tools-> Services). Open the mentioned "Services" dialog and search "Pandora Service", then click the button to stop it.

To uninstall the Pandora FMS Windows Agent, execute this sentence in a Windows command line:

```
PandoraService.exe --uninstall
```

#### 4.5.4 Windows Agent testing

You can check the Pandora FMS Windows Agent output in the *C:\pandora\pandora-debug.dbg* file, which is a plain text file and includes info about the execution flow of the Agent.

To test that SSH is working correctly, you can use the *--test-ssh* parameter with the executable file. This forces Pandora FMS to connect using internal SSH and copy a file named *ssh.test*.

#### 4.5.5 Pandora FMS Windows Agent configuration

All setup is made in *pandora\_agent.conf*. This file is a list of keys/values pairs. Here is an example of this file.

```
# General Parameters
# =====

server_ip 127.0.0.1
server_path /var/spool/pandora/data_in
temporal "c:\windows\temp"
interval 300
agent_name localhost

# Module Definition
# =====

# Counting OpenedConnections (check the language string)
module_begin
module_name OpenNetConnections
```

```
module_type generic_data
module_exec netstat -na | grep ESTAB | wc -l | tr -d " "
module_description Conexiones abiertas (interval 2)
module_interval 2
module_end

# Is Schedule service running ?
module_begin
module_name ServicioProg
module_type generic_proc
module_service Schedule
module_description Servicio Programador de tareas
module_end

# Is Eventlog service running ?
module_begin
module_name ServicioReg
module_type generic_proc
module_service Eventlog
module_description Servicio Registro de sucesos
module_end

# Is lsass.exe process alive ?
module_begin
module_name Proc_lsass
module_type generic_proc
module_proc lsass.exe
module_description LSASS.exe process.
module_end

# Received packets.
# Please notice that "Paquetes recibidos" string must be
# replaced by
# the correct string in your Windows system language.
module_begin
module_name ReceivedPackets
module_type generic_data
module_exec netstat -s | grep "Paquetes recibidos " |
    tr -d " " | cut -f 2 -d "=" | tr -d "\n"
module_description Conexiones abiertas (interval 2)
module_end

# Free space on disk
module_begin
module_name FreeDiskC
module_type generic_data
module_freedisk C:
module_description Free space on drive C:
module_end

# CPU usage percentage
module_begin
module_name CPUUse0
module_type generic_data
```

```
module_cpuusage 0
module_description CPU#0 usage
module_end

module_begin
module_name FreeMemory
module_type generic_data
module_freememory
module_description Amount of free memory.
module_end
```

#### 4.5.6 Agent visualizaton in Pandora FMS Web Console

Once Pandora FMS Agent is installed in the system to be monitored, you have to add the agent in Pandora FMS Web Console to see the data gathered.

To add the agent, go to the section «Manage Agents» from the «Administration» menu, and click on «Create agent», at the right bottom side. Insert the needed info about the agent, taking special care of the name, which **has to be the exactly the same of the physical agent**, being usually the host name where the agent was installed, and also the IP of the machine to monitor. Insert your wished *Group*, *Interval*, *OS*, *Server* and *Description*. The field *OS* is optional, and if you leave the description empty it will be autofilled with "Physical agent".

Once the Agent is created, you must wait some minutes before Pandora FMS Server processes the information sent, and then show it in the Web Console.

Like the rest of the agents, you can see its details at Operation -> View Agents -> Agent Detail, clicking on the name of the physical agent. For more details see [Pandora FMS basic use](#).

## 4.6 DATA TRANSFER TO SERVER

---

Once the agent is installed, configured and registered in Pandora FMS Web Console, it will start sending data to Pandora FMS Server, and it will be shown in the Console. To do so, there are several methods:

- **Tentacle**
- **SSH**
- **FTP**

### 4.6.1 Data transfer using Tentacle

**Tentacle** is a client/server file transfer protocol that aims to be:

- Secure by design.
- Easy to use.
- Versatile and cross-platform.

Right now is the preferred method, since Tentacle has been developed to replace more complex file transfer tools such as SCP/SSH and FTP, and switch from authentication mechanisms like *.netrc*. Simple password authentication over a SSL secured connection is supported too.

To learn how to install Tentacle on your client, read the [installation guide from the Tentacle User Guide](#).

To learn how to transfer files with Tentacle, read the [samples of usage in the Tentacle User Guide](#).

#### 4.6.2 Data transfer using SSH

The package data transfers through SSH can be done with DSA or RSA authentication.

The process is completely secure, since passwords don't travel over the net, nor unencrypted confidential data, so confidentiality, integrity and connection authentication between the agent and the server are assured.

To send the data files securely, you have to generate SSH keys, the process is detailed below, for both UNIX and Windows systems.

##### OpenSSH2 Key generation for UNIX

The SSH keys generated must be:

- *SSH* version2 keys
- *Open SSH* format keys
- *DiffieHellman* (DH) format keys

Due that Pandora FMS agent connects by SSH you need to setup SSH keys now. You also can use FTP method using a *.netrc* file, but **it is much more secure and a better option to use SFTP with SSH2**.

Probably, you want to run Pandora FMS agent under root privileges to grab system data. It is possible that you don't need to run it as root to collect data you need, in that case, procedure is the same, but with another user.

Create ssh keys using DSA type for key:

```
ssh-keygen -t dsa
```

And reply as follows to questions (enter to all questions):

```
Generating public/private dsa key pair.  
Enter file in which to save the key (/root/.ssh/id_dsa):  
Created directory '/root/.ssh'.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /root/.ssh/id_dsa.  
Your public key has been saved in /root/.ssh/id_dsa.pub.  
The key fingerprint is:  
xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:2d:68:30:f7:53:2d:7e
```

You have to add your PUBLIC key (*/root/.ssh/id\_dsa.pub*) to */home/pandora/.ssh/authorized\_keys* file in each Pandora FMS Data Server you want to use with this agent.

Login to Pandora FMS Data Server, and add the key to */home/pandora/.ssh/authorized\_keys* file. You can use cut & paste, for example, or copy the file with *scp* or *ftp* from one system to another. Watch out of carriage returns. The public key appearance is something like:

```
ssh-dss
AAAAB3NzaC1kc3MAAACBAMR4W00vuT3UyZPKC/NcqBudub/H8oKF2LRv52LX88YNokgdIPNOa
tNeweCuQdVOaDUNvFTgnyYV6iBtApstzU16ndKALZ1DoZnBYULYTUtBcdRHq7vn0bufIMRHFp
g8ZvqR3dBulz6bVQqJu8nqZGQDyLgPEmkQ6O9 root@blackbox01
```

**The entire block MUST BE in a SINGLE LINE**, if not, it will not work. Also, */home/pandora/.ssh/* directory and */home/pandora/.ssh/authorized\_keys* in the server, must be owned by "pandora" user with its permissions set to 700 for the directory, and 600 for the file *authorized\_keys*.

For example, if you copied *id\_dsa.pub* to */tmp* in server system:

```
cat /tmp/id_dsa.pub >> /home/pandora/.ssh/authorized_keys
chmod 600 /home/pandora/.ssh/authorized_keys
chmod 700 /home/pandora/.ssh/
chown -R pandora /home/pandora/
```



**Warning!** Setting up SSH authentication is hard, any missed step will result in a fail, so please don't skip anything.

Always test this connection to check that SSH authentication is working. From your system agent, where the Pandora FMS Agent is running, try to contact Pandora FMS server:

```
ssh pandora@server_ip
```

The first time a hostkey authentication changes, it should show you something like this:

```
The authenticity of host 'xxxxx (x.x.x.x)' can't be established.
RSA key fingerprint is 42:d4:a5:f2:a7:b8:1f:c3:d5:42:ab:c7:b5:5b:af:57.
Are you sure you want to continue connecting (yes/no)?
```

Reply "yes", and you should see the system prompt for "pandora" user, **WITHOUT** being asked for a password, because SSH automatic authentication, based on DSA Keys, should work and resolve authentication. If not, try to review previous steps. Note that if you're using *scponly* protection, shell won't be available, but authentication will be correct.

If you have serious problems and get stuck, try to setup maximum verbosity for SSH Daemon on the system which is running Pandora FMS server:

```
vi /etc/ssh/sshd_config
```

Replace *LogLevel INFO* with *LogLevel DEBUG2*

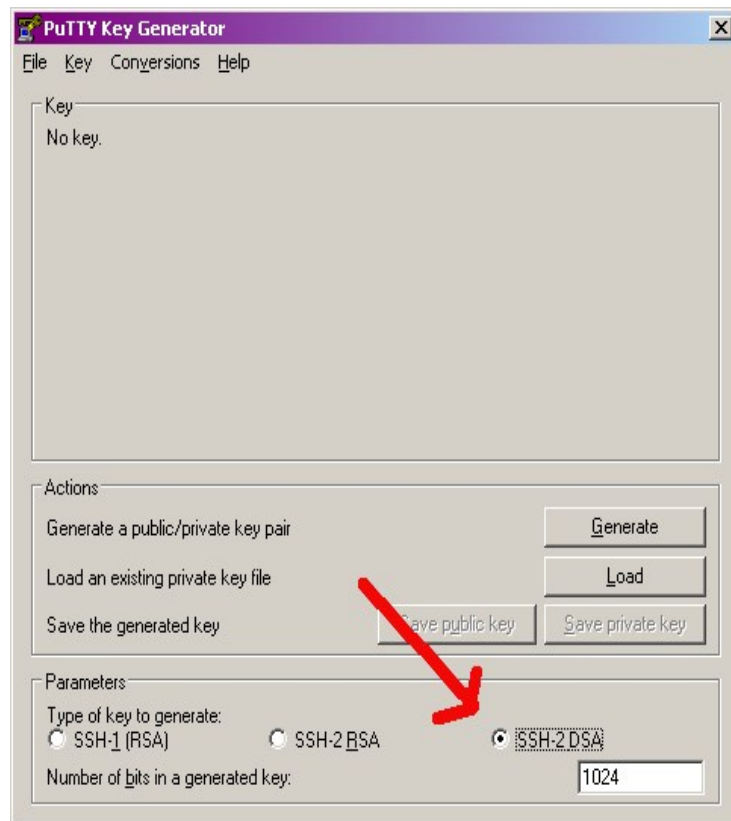
Restart SSH service:

```
/etc/init.d/ssh restart
```

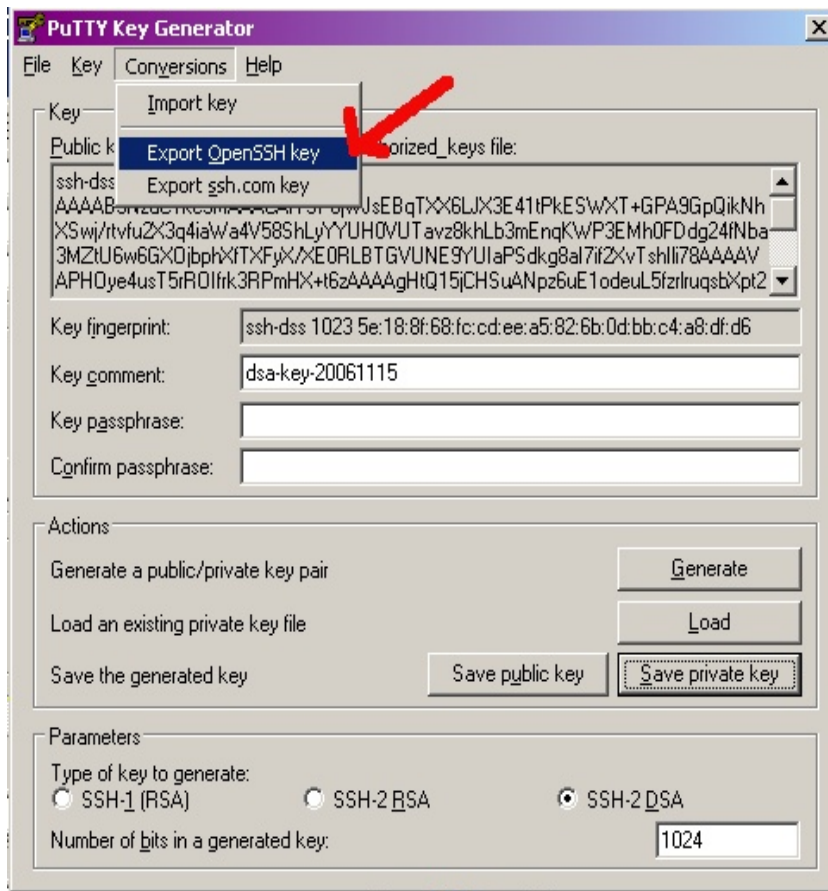
**Now you have much more information about SSH connections.** In Debian this information is at */var/log/auth.log*. Don't forget to set again *LogLevel INFO* in your *sshd\_config* file, and restart again SSH, or too much login messages will be generated (with performance penalty).

### 4.6.3 SSH Key generation for Windows

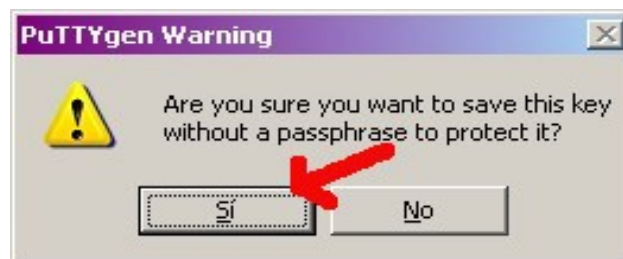
Go to the *.\util* directory of your Pandora FMS Windows Agent and run *puttygen.exe*. Choose the option "*Generate keys, SSH-2\_DSA, 1024*".



Press "*Generate*". Export the key as *OpenSSH* (Pandora FMS SSH implementation uses OpenSSH port).

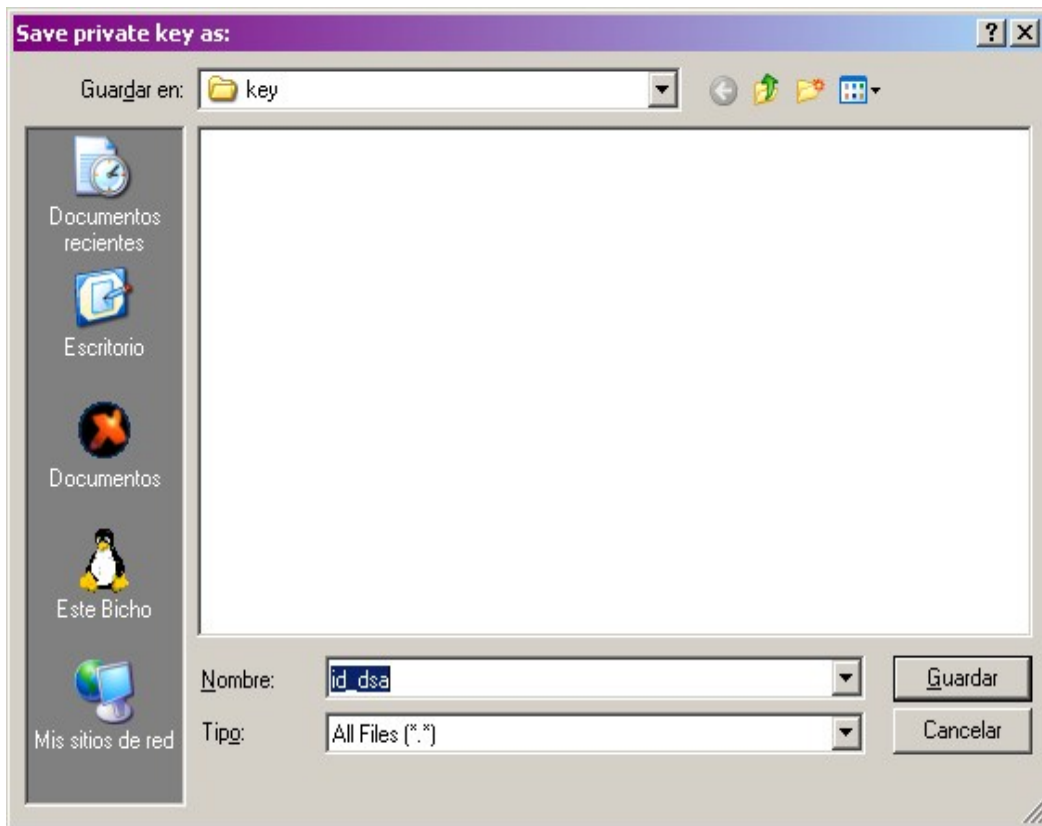


We didn't choose any password, so press "YES":

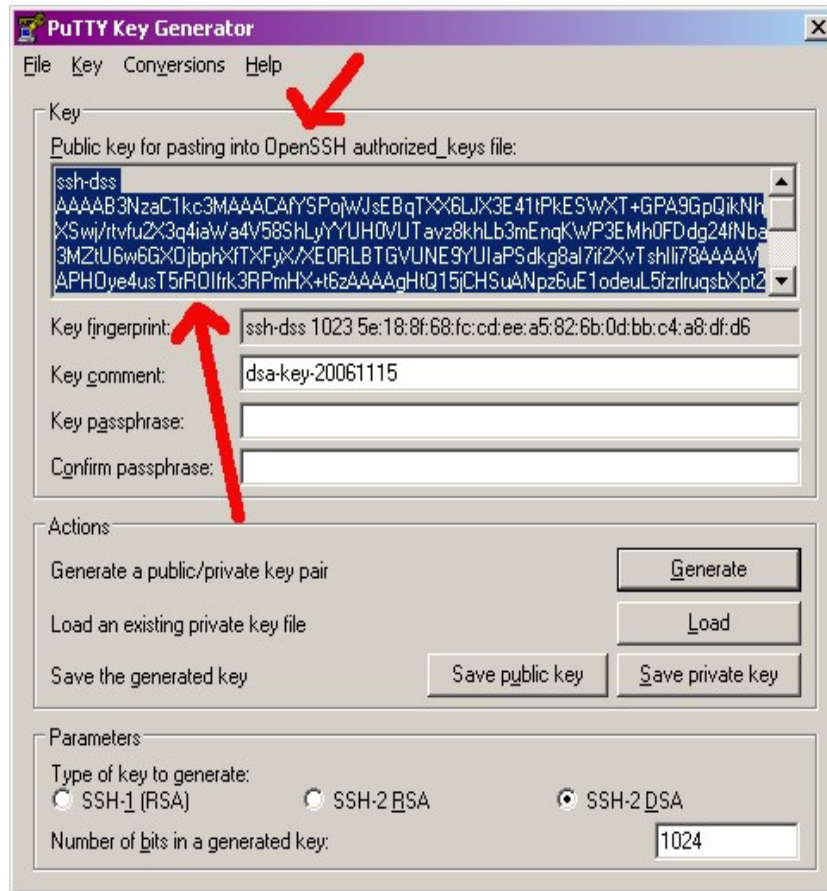




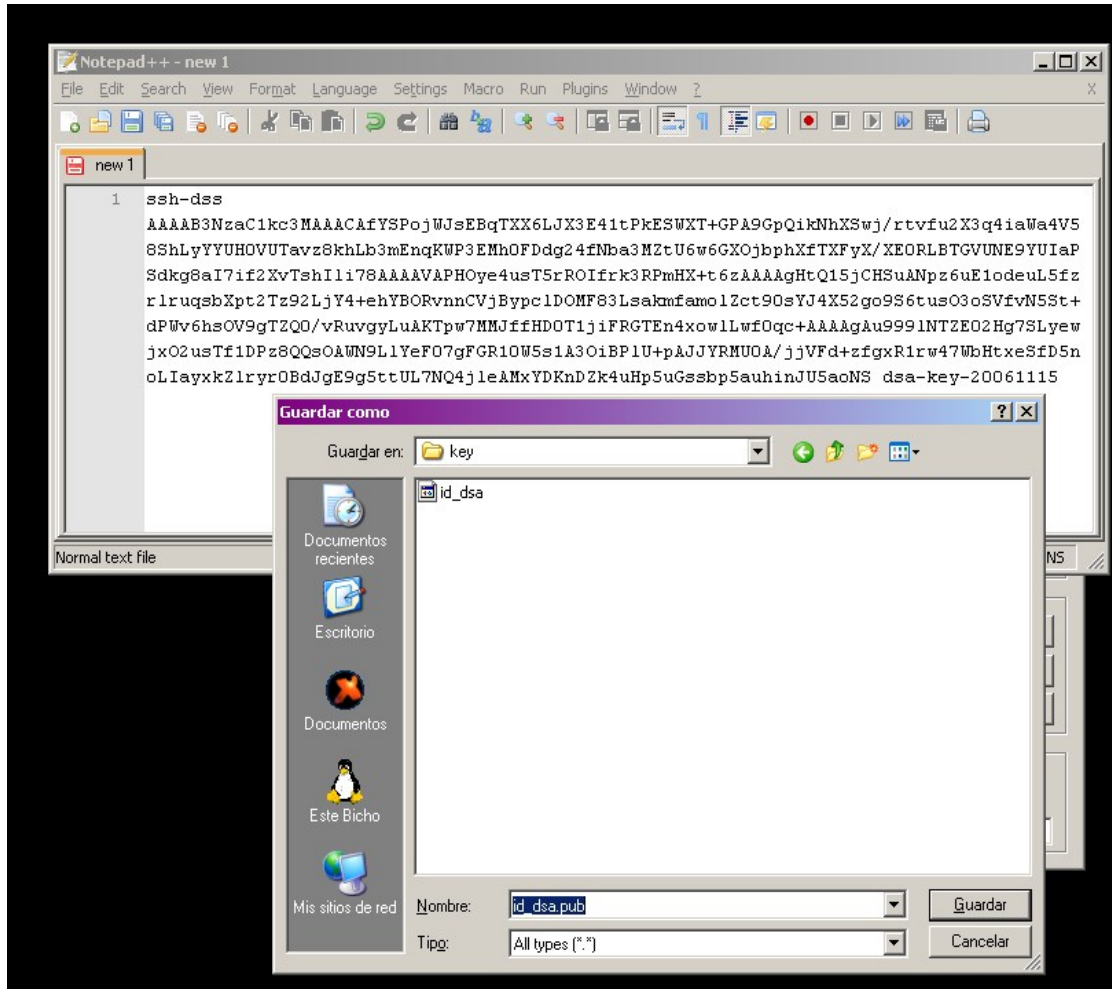
*Save it as C:\Program Files\Pandora\_Agent\keys\id\_dsa:*



Now copy the public key to clipboard:



and paste it as *C:\Program Files\Pandora\_Agent\keys\id\_dsa.pub*, and also to */home/pandora/.ssh/authorized\_keys* file in the server to establish a correct SSH automatic key authentication.



#### 4.6.4 Data transfer using FTP

You need to have a FTP client and server installed in the machines from which you want to send/receive, respectively, and the process is the standard one for file transferring using FTP, automatic once configured the option at the agent configuration file.

## 5 BASIC MONITORING

### 5.1 PANDORA FMS SERVERS

Agents defined in Pandora FMS Web Console can contain different type of modules. In Pandora FMS v1.3 they can contain up to three module types (by the way they process the information and its source):

- **Network modules**, processed by the Network Server.
- **Data modules**, processed by the Data Server.
- **Keepalive module**, that is enabled only when it doesn't receive any data from the modules, of any kind, for a period two times the defined one. This module is generated by any of the active servers.

By the way Pandora FMS works, the alerts are executed by those servers that process the module. Which means that the execution of the script or command is performed in the system where the server that processes the data is running: if it is a network module, it would be a Network Server, if it is a data module that comes from an agent, then it would be a Data server. In case you have server redundancy, it will be the server which is processing the data.

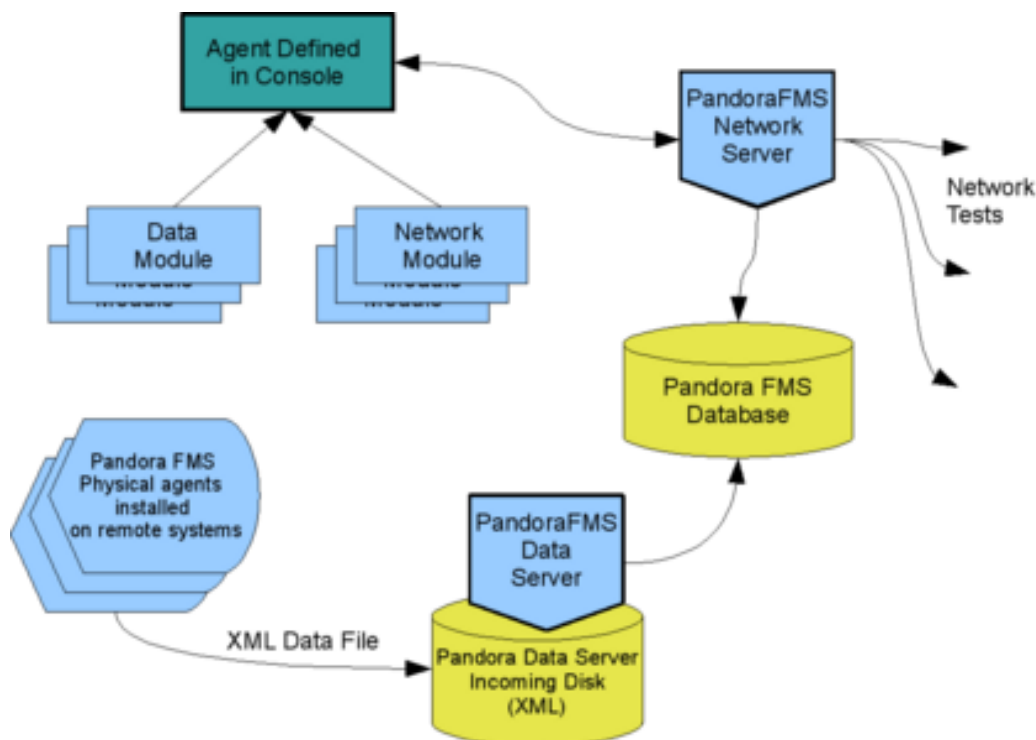


Figure 4: Global schema of Pandora FMS Servers

## 5.2 NETWORK SERVER

---

Pandora FMS Network Server is a key piece since it allows to perform remote and centralized tests. on the contrary to the Data Server, the Network Server executes its assigned tasks using a multiprocess queues system. A Network Server can also work with other servers to balance the network load, and act as backup in case any fails down, taking over the work of the server that went down. To learn more about HA in Pandora FMS, read the chapter [Monitoring with Pandora FMS: Advanced setup](#).

The Network Server only works with those network modules assigned to it. Obviously, the Network Server has to have a whole vision of (IP adresses and ports) of the systems to monitor. Is has no sense to perform network tests against a system which ports can't be seen or from which routes are unknown. The existence of firewalls has nothing to do with Pandora FMS and the problems derivated from its use aren't also related with any specific Pandora FMS configuration.

### 5.2.1 Network Modules

Pandora FMS network modules perform remote monitoring tasks. Those remote monitoring tasks can be summarized into three blocks:

- ICMP tests: If a machine replies to a *Ping* (*remote\_icmp\_proc*) or a system latency time (*remote\_icmp*). Both tests are performed by the Network Server which has the agent containing those network modules.
- TCP tests: Remotely you can confirm that some system has a specified port open, and the port to check can be set at the module definition. A text string can also be sent (using the string "^M" to replace the carriage return), and check that there is a successful communication by its answer. This allows to implement simple protocol checks. For example, we could check wether a server is "alive" sending the string *GET/HTTP/1.0^M^M* and receiving the string "200 OK".
- SNMP tests: It is possible to launch remote SNMP requests (*SNMP Polling*) to systems which SNMP service is activated and accessible to get data from them, such as the status of the interfaces, the network traffic, and so on. There is a section about SNMP and Pandora FMS (see below).

The Network Server is the piece that performs the different network tests assigned to each agent. Each agent is assigned to a Network Server, the one in charge of its execution, inserting the data into de Database of Pandora FMS system.

Componente de red	---Configuración manual---		Actualizar
Nombre módulo		IP destino	127.0.0.1
Tipo módulo	--	Grupo módulos	General
Intervalo módulo	300	Puerto TCP	
SNMP OID		Comunidad SNMP	public
SNMP OID			SNMP Walk →

Figure 5: Doing a "SNMP walk" over a SNMP device

## 5.2.2 Pandora FMS and SNMP

Pandora FMS can work with any device that supports SNMP. Although Pandora FMS only works with SNMP v1, it can be easily modified to work with SNMP v2 and v3.

Pandora FMS works with SNMP handling individual OID. Pandora FMS understands each OID as a network module. Which means that if you want to monitor a 24 port Cisco Catalyst switch, and know the status of each port as well as the incoming and outgoing traffic, you have to define up to 72 modules (24 x 3).

To work with SNMP devices you need to:

- Learn what is and how works SNMP protocol. This is described at [RFC3411](#), published by the IETF.
- Know the IP and SNMP community of the remote device.

### FORMULARIO DE ASOCIACIÓN DE MÓDULO

Componente de red	---Configuración manual---		Actualizar
Nombre módulo		IP destino	192.168.50.250
Tipo módulo	--	Grupo módulos	General
Intervalo módulo	300	Puerto TCP	
SNMP OID		Comunidad SNMP	default
SNMP OID	SNMPv2-MIB::sysDescr.0 - Cisco Internetwork O SNMPv2-MIB::sysDescr.0 - Cisco Internetwork O SNMPv2-MIB::sysObjectID.0 - SNMPv2-SMI::enterpri DISMAN-EVENT-MIB::sysUpTimeInstance - 52.0:15:15.02 SNMPv2-MIB::sysContact.0 - SNMPv2-MIB::sysName.0 - almendra SNMPv2-MIB::sysLocation.0 - SNMPv2-MIB::sysServices.0 - 2 SNMPv2-MIB::sysORLastChange.0 - 0:0:00:00:00 IF-MIB::ifNumber.0 - 26		SNMP Walk →
Enviar TCP			

Figure 6: Choosing an element for a SNMP device over which the "SNMP walk" was performed

- Activate the device SNMP management so that from the Network Server you can perform SNMP queries. This Network Server must be the one that assigned by the agent where the network modules are defined. It is also important to take into account that if you want other servers to query the device in case the assigned one fails, these ones will do the queries with another IP address.
- Know the specific OID of the remote device to query.
- Know how to handle the data returned from the device. SNMP devices return data

in different formats. Pandora FMS can work with almost all of them, except *timetick* which is handled as a numeric format without changing them into date/time. Counter data types are the ones managed as *remote\_snmp\_inc* by Pandora FMS, and are of a special importance, since they cannot be handled as numeric data. Most of statistical SNMP data are counters and they have to be configured as *remote\_snmp\_inc* if you want to monitor them correctly.

To avoid defining again and again the same modules, some tools were created: *network components*, *network templates* and the configuration copy tool.

Pandora FMS has also a simple SNMP browser that allows to walk the MIB of a remote device by a *SNMP Walk*.

If you want Pandora FMS to walk and browse remote SNMP devices, you need to install the package **NetSNMP** as well as **SNMP PHP5 extensions**.

You may also want to install new MIB (sets of SNMP trees), then you will have to do it as specified at NetSNMP manual, which usually consists of copying the MIB files to */usr/share/snmp/mibs* of Pandora FMS Web Console. To learn more about how does MIB work under GNU/Linux, you can check the project documentation at [NetSNMP web site](#).

#### Common OIDs to use with Pandora FMS

This section describes a set of typical OIDs to Pandora FMS use with devices that accept SNMP. To know more specific OIDs it would be necessary to contact the manufacturer, since depending on them there are comprehensive MIB trees with multiple values for each device, completely different among manufacturers, and even between models of the same device. There is a section explaining how to work with external SNMP handlers, and how to work with MIB externally to Pandora FMS.

Network statistics by interface (supported by any device with SNMP implementation):

- IF-MIB::ifDescr.[N° int]: Interface name (eth1, if0...), as a text string.
- IF-MIB::ifInOctets.[N° int]: Input traffic (bytes) incremental.
- IF-MIB::ifOutOctets.[N° int]: Output traffic (bytes) incremental.
- IF-MIB::ifOperStatus.[N° int]: Interface status (boolean-special, handled with *\_proc* types in Pandora FMS).

With systems that implement the system information MIB UCD-SNMP (several UNIX and GNU/Linux):

- UCD-SNMP-MIB::memAvailReal.0: Free memory.
- UCD-SNMP-MIB::memTotalReal.0: Total memory.
- UCD-SNMP-MIB::memAvailSwap.0: Free *swap*.
- UCD-SNMP-MIB::memTotalSwap.0: Total *swap*.

You can also know the CPU load with the following OIDs:

- UCD-SNMP-MIB::laLoadInt.1: Avg. load / 1 min.
- UCD-SNMP-MIB::laLoadInt.2: Avg. load / 5 min.
- UCD-SNMP-MIB::laLoadInt.3: Avg. load / 15 min.

It is also possible to know the CPU usage (%):

- UCD-SNMP-MIB::ssCpuRawUser.0 : Number of timeticks used by users's process (AIX, Solaris, Linux, HP-UX).
- UCD-SNMP-MIB::ssCpuRawSystem.0 : Number of timeticks used by system's process (AIX, Solaris, Linux, HP-UX).
- UCD-SNMP-MIB::ssCpuRawIdle.0 : Number of timeticks unused (AIX, Solaris, Linux, HP-UX).
- UCD-SNMP-MIB::ssCpuRawWait.0 : Number of timeticks used by process waiting (AIX, Solaris, HP-UX).
- UCD-SNMP-MIB::ssCpuRawNice.0 : Number of timeticks used by nice process (Linux, HP-UX).
- UCD-SNMP-MIB::ssCpuRawKernel.0 : Number of timeticks used by kernel process (AIX, Solaris, Linux, HP-UX).

Number of processes and users:

- HOST-RESOURCES-MIB::hrSystemNumUsers.0: Number system users.
- HOST-RESOURCES-MIB::hrSystemProcesses.0: Number os system processes.

#### **MIB study with external tools and integration with Pandora FMS**

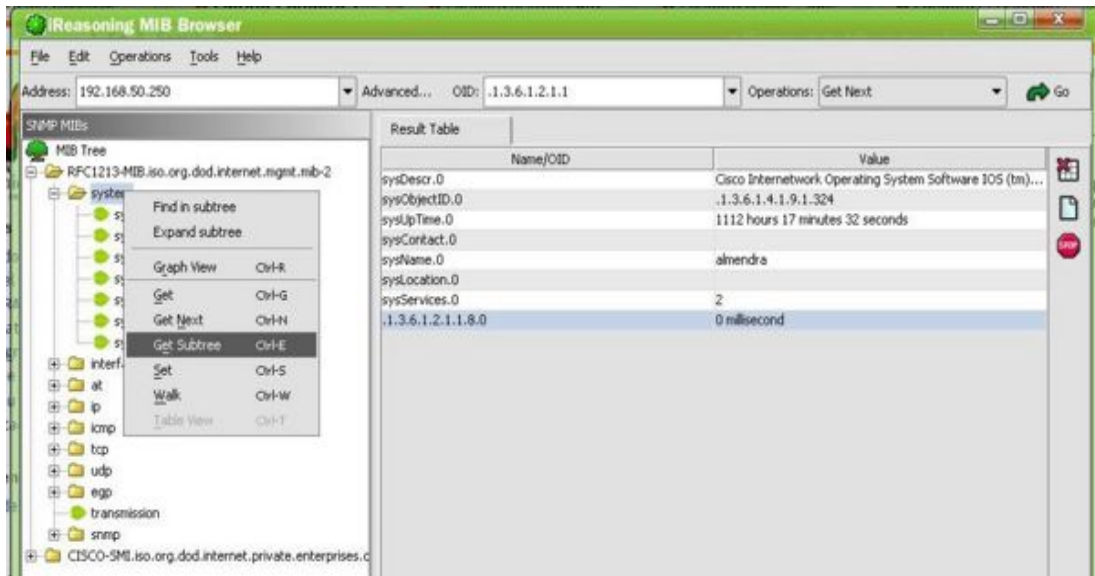
We recommend to use a MIB browser to to do an analysis of the available OIDs to use with Pandora FMS, and analyze the offered MIB by each manufacturer. These MIB browsers are desktop tools that read, process, analyze and show the complete OID tree of each MIB, allowing searches and let you know which OID are necessary to monitor your devices.

The following external MIB handling tools are recommended:

- iReasoning MIB Browser (Windows, Linux, Java): [\[1\]](#)
- Get-If Free MIB Browser (Windows): [\[2\]](#)
- TKMib: for UNIX, standard in most of GNU/Linux distributions.

The following screenshots are taken from iReasoning MIB Browser tool.





At the first screenshot you can see a request of the device with a loaded MIB (*MIB2 default*) that recognises some of the OID present. These OID can be represented as a string or using a numeric format. Pandora FMS understands both, but it can only resolve alphanumeric OID, if it has the correct MIB loaded in the Operating System. The best is to use numeric OID.

At the second screenshot you can see the result of a recurse *SNMP Walk* over a branch from which we do not have MIBs, so we get several numeric OID that serve no purpose, since there is no info about them, or the kind of data they provide.

.1.3.6.1.2.1.1.8.0	0 millisecond
.1.3.6.1.4.1.9.2.1.1.0	Bootstrap program is C2950 boot loader
.1.3.6.1.4.1.9.2.1.2.0	power-on
.1.3.6.1.4.1.9.2.1.3.0	almendra
.1.3.6.1.4.1.9.2.1.4.0	
.1.3.6.1.4.1.9.2.1.5.0	0.0.0.0
.1.3.6.1.4.1.9.2.1.6.0	0.0.0.0
.1.3.6.1.4.1.9.2.1.8.0	5132512

Outside anything you can do with a MIB exploration tool, you can use OID references through OID indexes (some manufacturers provide MIBs and OIDs references), or by links that gather interesting OIDs. Other manufacturers provide SNMP stacks and document their SNMP catalogs with a natural language, easy to understand and to get the OID needed (for example, UCD-SNMP, the SNMP stack used by most UNIX systems). Many other Operating System SNMP stacks, such as AIX or Windows, are also very well documented.

### Recommended links to work with SNMP

- **Full OID Catalog for CISCO** (extremely useful): [\[3\]](#)

- HP Printer MIB: [\[4\]](#)
- Nagios Exchange - SNMP [\[5\]](#)
- Some SNMP OID frequently used by *routers*: [\[6\]](#)

### 5.2.3 Network profiles

With Pandora 1.3 two new feature have been added: *Network Components* and *Network Components Templates/Profiles*.

A **network component** is a "generic" network module that can be **reused** many times. It is defined like another network component, and placed on a library or group of network components to be used, assigned to an agent individually or using a template called **Network Profile**.

To edit or create a network component, go to Administration -> Manage Modules -> Network Components.

#### GESTIÓN DE COMPONENTES DE RED

Nombre módulo	<input type="text" value="NIC #1 status"/>	Tipo módulo	<input type="text" value="remote_snmp_proc"/>
Grupo	<input type="text" value="Network Management"/>	Grupo módulos	<input type="text" value="Networking"/>
Intervalo módulo	<input type="text" value="0"/>	Puerto TCP	<input type="text"/>
SNMP OID	<input type="text" value=".1.3.6.1.2.1.2.2.1.8.1"/>	Comunidad SNMP	<input type="text" value="public"/>
Enviar TCP	<input type="text"/>	Recibir TCP	<input type="text"/>
Dato mín.	<input type="text" value="0"/>	Dato máx.	<input type="text" value="0"/>
Comentarios	<input type="text" value="Estado de NIC #1"/>		

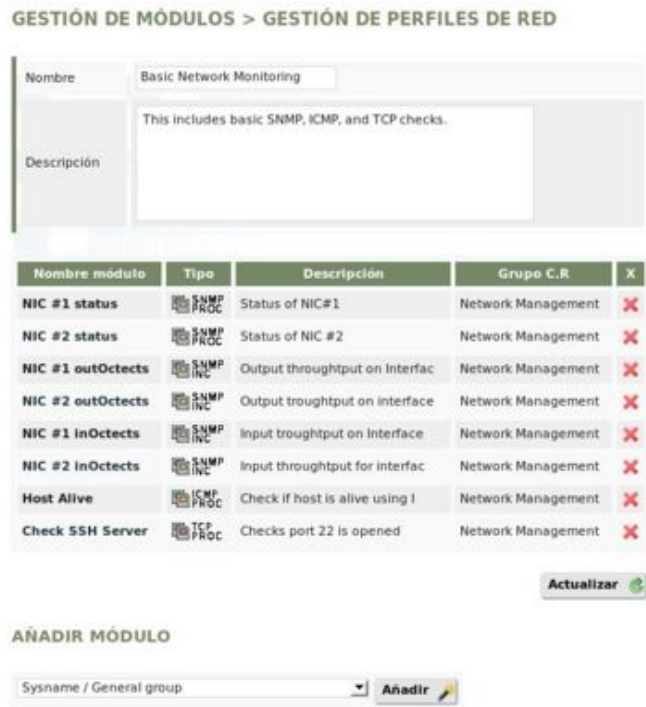
Figure 7: Defining a network component

This is similar to a standard module definition, except that you add new group (for "Network Components") and you don't have the field IP Address. This is because the IP Address field will be replaced with primary IP Addresses of the agents which were assigned to be owners of this network component. This network component creates a standard network module with the same data that contains the network component.

A network profile, or network template, is a collection of network components that can be assigned to a *recon* task, or to an agent in wizard mode.

To edit network profiles, go to Administration -> Manage Modules -> Network Templates.

This simply contains a group of network components. You can create as many as you want. Its typical use is to create a profile for each "server type" you have in your network, and several "basic" profiles for monitoring network connectivity using a *ICMP Proc check* (a simple ping).



### 5.3 RECON SERVER

Figure 8: Defining network component

Pandora FMS Recon server was introduced in 1.3 version. It recognizes the network using user-defined tasks, to find new systems (identified by an IP address) and add to monitor, using the Network Profiles to assign automatically modules to the new agent. It is important to remark that it uses IP addresses to identify agents already monitored by Pandora FMS, this is the reason why from 1.3 version, agents can have more than one IP address.

Recon tasks are defined in Administration Menu -> Manage Servers -> Manage Recon

#### DETALLE DE CONFIGURACIÓN - NIMROD\_RECON

	Nombre tarea	Intervalo	Red	Tipo	Estado	Perfil de red	Grupo	Progreso	Actualizado el	
⊕	Descubrir equipos	~ 1 horas	192.168.50.0/24	ICMP	Pendiente	Basic Network Monitoring	🟢	<div style="width: 49%;"><div style="width: 49%;">49%</div></div>	03/19/08 13:45:19	🔧
⊕	Descubrir equipos 2	~ 1 horas	192.168.50.0/24	ICMP	Pendiente	Basic Monitoring	🟢	<div style="width: 64%;"><div style="width: 64%;">64%</div></div>	03/18/08 16:50:18	🔧

Figure 9: Recon task status

task.

Before defining a new task, you need to start a Recon Server in order to be detected in system as recon server. To assign new agents automatically to a Network Server, you also need to start a **Network Server**.

Network targets for the scan, are defined using [CIDR](#) format. For example 192.168.1.0/24 refers to "all C class on 192.168.1.0".

On this version (1.3) only ICMP scanning are possible. For next versions we are planning to implement scanning specific ports, and specific text response with SNMP queries.

## SERVIDORES PANDORA > GESTIÓN RECONTASK

Nombre tarea	<input type="text"/>	Servidor Recon	<input type="text"/>
Red	<input type="text"/>	Intervalo	0.5días
Perfil de red	Basic Network Monitoring	Servidor de red	<input type="text"/>
Grupo	All	Tipo	ICMP
Incidente	Sí		
Comentarios	<input type="text"/>		

Figure 10: Defining a recon task

Group and network profile assigned to new hosts in this sweep allow to acquire a detailed view of huge networks and deploy a network monitoring for it in minutes or hours. You can detect and begin monitoring your whole network in a few steps.

### 5.3.1 Example of use

If you have four C classes for Network Servers, and a B class with many workstations, you can define a Network Profile for each of that five networks. For example:

- Profile number #1: Used for Windows server. This could have five modules:
  - SNMP module to get CPU usage on Windows server.
  - SNMP module to get available memory on Windows server.
  - SNMP to get network interface incoming throughput.
  - SNMP to get network interface outgoing throughput.
  - ICMP check to see if it is alive.
- Profile number #2: Used to check UNIX HTTP servers:
  - ICMP check to see if it's alive.
  - TCP check to see if port 80 is alive and responding to HTTP commands.
  - TCP check to see if port 22 is alive and responding to SSH.
  - SNMP to get CPU usage.
  - SNMP to get network interface incoming throughput.
  - SNMP to get network interface outgoing throughput.
- Profile number #3: Used to check UNIX Oracle servers:
  - ICMP check to see if it is alive.
  - TCP check to see if a specific TCP port is alive and responding to Oracle commands.

- TCP check to see if a specific port is opened.
- SNMP to get CPU usage.
- SNMP module to get available memory.
- Profile number #4: Used to check Windows CIFS servers:
  - ICMP check to see if it is alive.
  - SNMP to get CPU usage.
  - SNMP module to get available memory.
  - Several TCP check to verify CIFS availability.
  - SNMP to get network interface incoming throughput.
  - SNMP to get network interface outgoing throughput.

Profile number #5: Used to check all workstations activity:

- ICMP check to see if it is alive.
- TCP check to check that specific "forbidden" ports are closed, like 21, 22, 80, 8080, 5900, P2P ports, etc.

Create five recon tasks, four for each kind of server in each network or subnetwork assigned to this kind of servers. Assign each task to a different group and assign its network profile. The last one, for the workstation, assign it to the whole B class and to a different group. Use a shorter interval of scan (half day, one day) for workstations and longer for servers (2-3 days or a week).

Recon servers use a internal ICMP scan to check whether the machines are alive or not. When they create an agent, they try to resolve IP address to set the hostname as the name of the agent.

## 5.4 PANDORA FMS AGENTS

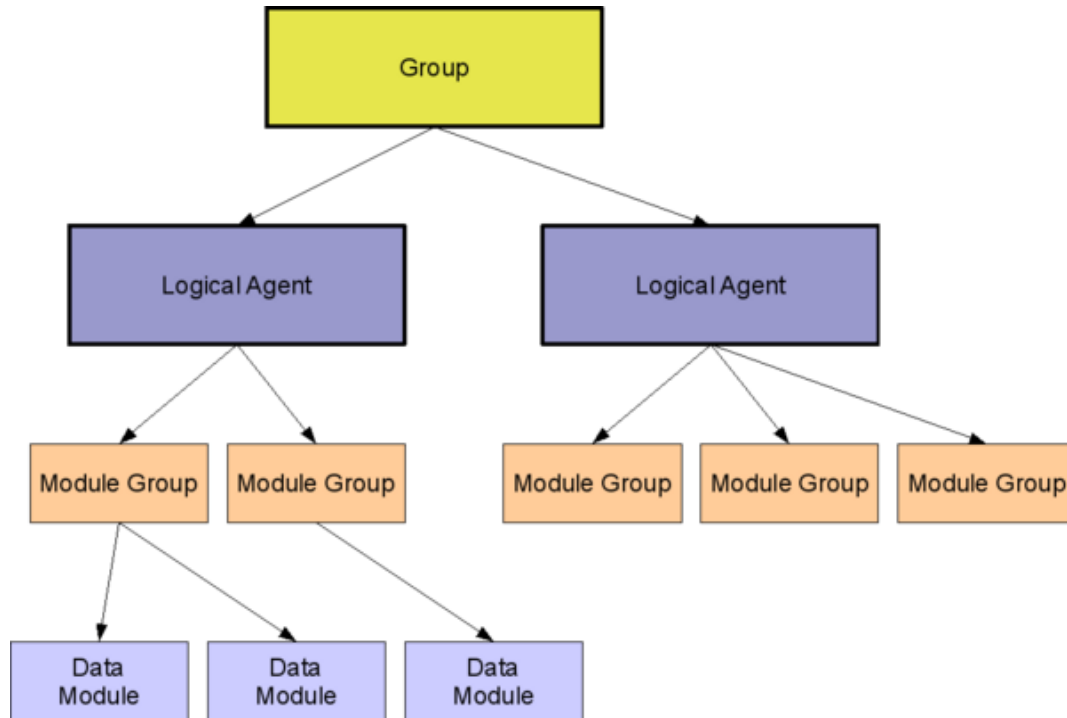


Figure 11: Logical information hierarchy in Pandora FMS

All monitoring tasks performed by Pandora FMS is done through the agents. The information is sorted logically by a simple hierarchy based on groups, agents, module groups and modules. There are agents based just in the information provided by a software agent, and installed on the System, and agents with network information only, information that does not come from any physical agent, where there is no need to install any software, and that performs network monitoring tasks from Pandora FMS Network Servers. The information is received in modules, logically assigned to Pandora FMS agents. It is important to distinguish the concept of logical agent (which inside has modules that contain the gathered information), and the agents as software pieces that run in remote hosts.

### 5.4.1 Pandora FMS physical agents

The data gathered by agents is stored in small pieces of information called "modules". Each module stores only one kind of data. The value of each module is the value of the supervised variable. The agent will be created with the same name that the remote agent has in Pandora FMS Web Console. Then, the data will start to be inserted in the Database and you will be able to access to them. **It is very important to check that the name of the agent is the same as the agent defined in the Console.**

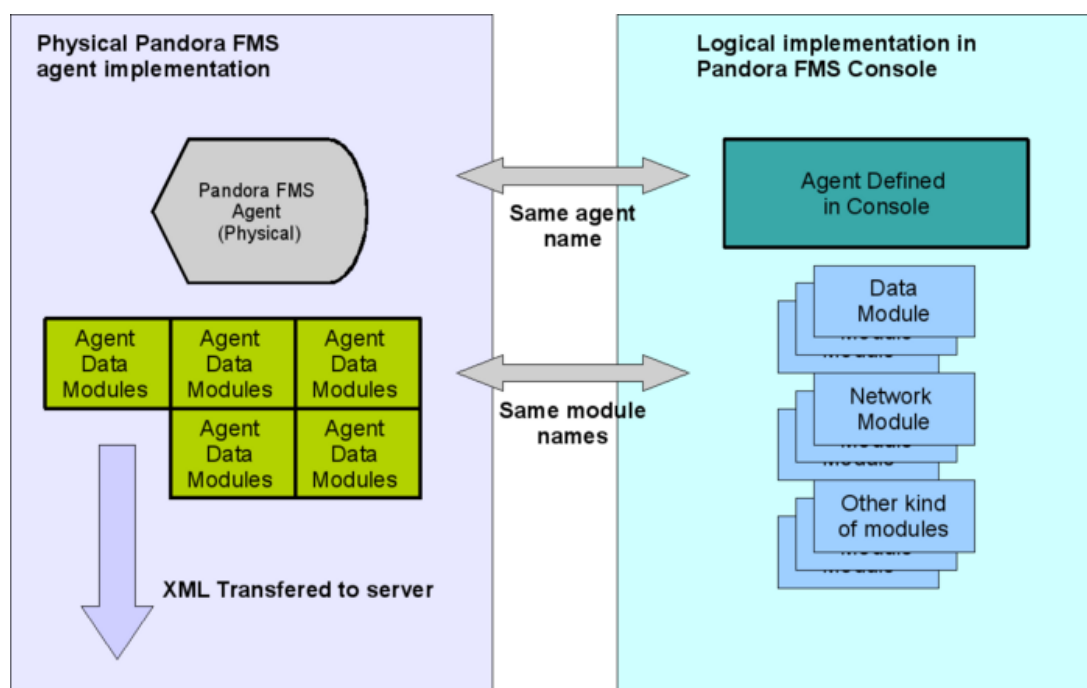


Figure 12: Pandora FMS Agents logical scheme

Read the section [Installing and using Pandora FMS Physical Agents](#) to learn more about them.

#### 5.4.2 Network agents / Network modules

A network agent needs to be assigned to a Network Server to perform remote monitoring tasks. If you don't see any Network Server, then there isn't any Network Server running. You will have to set up and start one before trying to assign a network module to an agent. You can also assign network modules to a logic agent that already contains data modules of the "data server" type. A logic agent can contain information that comes from remote monitoring network modules and information modules that come from an agent through the "Data Server".

#### 5.4.3 Assigning modules to a Pandora FMS agent

Pandora FMS software agents use the operating system own commands to get the information. Pandora FMS Data Server stores and processes the output generated by those commands, transferred to the server inside a XML file. The info returned by those commands is what is called "module". If the agent was added in "learning mode", then those modules not defined previously in the logic agent, will be created automatically by the Server.

The network modules also perform tasks to get information, but since this monitoring is remote, these tasks are done from the network modules. You have to define those modules (or can be autocreated through an automated detection from a *Pandora FMS Recon Server*).

To create any kind of module, you have to follow the same steps. Go to the view "Manage Agents". Here you can see a list with all Pandora FMS agents. Choose an agent (or create one) and go to the tab "Modules". Here you can create a new module using the form at the end of the list of existing modules.

To add a module it is necessary to fill some of the following fields:

### Module type

There are several types, mainly classified in two: data that comes from software agents, and data that comes from network modules executed by a Network Server. Those identified as "generic" are modules with data that comes from software agents, and those identified as "remote" are network modules.

- **generic\_data**: numeric data type. It stores numeric data (integer and floating point ones) obtained from a module of an agent of Pandora FMS.
- **generic\_data\_inc**: incremental numerical data type. It stores data resulting from difference between last agent data and actual data. Pandora FMS Server calculates and stores the ratio automatically. Every module that ends with the suffix "inc" is of an incremental type.
- **generic\_data\_proc**: boolean data type: 0 means False or "bad value", and 1 or higher means True or "good value". "Generic Proc" types are also called "monitors" because they indicate whether something is "ok" or not without processing it, or without executing alerts. They are displayed as small lights at the agent view. Red if 0, Green if greater than zero. Every module that ends with the suffix "proc" is a monitor.
- **generic\_data\_string**: Alphanumeric data type (text string, max. 255 charas).
- **generic\_icmp**: get network latency (in miliseconds) remotely.
- **generic\_icmp\_proc**: execute a "ping" to remote system. Returns 0 if the system is not reachable or not responding. Returns 1 if the system could be reach. Alike "generic\_proc" this is a monitor.
- **remote\_tcp\_proc**: execute a "tcp ping" to remote system and returns "1" if the listed port is responding. Optionally, you can also send parameter inside the field "TCP SEND" and wait for the string defined at "TCP RECEIVE". If Pandora FMS Network Server receives the string defined at "TCP RECEIVE", then it returns 1 (ok), if not 0 (wrong). You can use the macro "^M" to send carriage returns. It is a monitor.
- **remote\_tcp\_data**: *remote\_tcp\_string, remote\_tcp\_inc, gets a numeric value from a TCP port. If it cannot connect, then it doesn't return any value. For string type it gets strings, and for tcp\_inc type, it gets a counter, like the equivalent members of the "generic" family.*
- **generic\_snmp (several)**: they get information using a SNMP interface. If you enter the SNMP community and IP address, you can get all the SNMP data from the destination using standard MIB stored in Pandora FMS Web Console using SNMP protocol v1. Walking (*SNMP Walk*) over a device will make all MIB variables available, so you can choose one. You can also enter a MIB using numerical OID or human understandable format, if you have the correct MIB installed in your Pandora FMS Network Server.



**Module Group:** it is possible to group modules inside the defined groups. You can also add more modules adding them it into Pandora FMS Database.

**Module Name:** we recommend using short names, since the name will be truncated in the views, making it harder to distinguish among them.

**Module Interval:** interval of time, in seconds, that the agent waits between two consecutive executions. If there are no data received from the module in the considered time (twice the module interval, or twice the agent's time if the first one is not defined), then Pandora FMS considers there is no response from that module.

**Target IP:** IP of the module. You can use [FQN](#) if Pandora FMS Network Server that is executing the network module is able to translate the name. An IP address can always be used.

**TCP port:** TCP port used by the network module.

**SNMP OID:** module's SNMP OID. If there is a MIB able to resolve the name in Pandora FMS Network Server, then you can use alphanumeric OIDs (i.e. SNMPv2-MIB::sysDescr.0). Numeric OID can always be used (i.e. 3.1.3.1.3.5.12.4.0.1), even if there is no specific MIB.

**SNMP Community:** Community needed to monitor a SNMP OID.

**TCP send:** Field to configure the parameters to send to the TCP port.

**TCP receive:** Field to configure the parameters which we expect to receive in a TCP connection.

**Maximum, Minimum:** max and min values for the module. Any value above/below this threshold will be taken as invalid and the whole module will be discarded.

#### 5.4.4 Creating a logic agent from a network module

You can define new agents from Pandora FMS Web Console. Once defined in Pandora FMS Web Console, it will be ready to receive data from a software agent (old agents, based on software installed in a remote machine), or from network modules (assigned to a Network Server that runs network tasks to monitor remote systems). You can also mix both types of module in the same logic agent.

Please remember that a network agent needs to be assigned to a Network Server to execute network tasks. If you cannot see any Network Servers it is because you don't have any Network Server running. Please configure and run a Network Server before trying to assign a network module to an agent.

To add a new agent the following parameters must be configured:

- **Agent Name:** Name of the agent. This and the "agent name" parameter in *pandora\_agent.conf* file must have the same value. The agent takes the hostname of the machine where it is running by default.
- **IP Address:** IP address of an agent. An agent can share its IP address with other agents, and have multiple IP addresses, but only one is the main "IP" address used

to set it as default in network modules. Pandora FMS Recon Server uses agent's ip address to know whether an agent is already being monitored or not.

- **Group:** the group of Pandora FMS where the agent belongs. In this version of Pandora FMS, an agent only can belong to a group.
- **Interval:** agents' execution interval. It is the elapsed time, in seconds, between two consecutive agent executions. An agent can have a defined interval, but there could be more modules with different (higher or smaller) intervals. An agent is considered "down" (not responding) when Pandora FMS servers (any of them) has no contact with it in for twice the interval time. The interval time is set in seconds.
- **OS:** The Operating System to be monitored. The supported Operating Systems are: AIX, BeOS, BSD, Cisco, HP-UX, GNU/Linux, MacOS, Other, Solaris, Windows. It is a optional parameter since monitoring does not take it into account.
- **Server:** in Pandora FMS 1.3.x you can assign here the Network Server assigned to the agents network modules. The normal data module not processed by the Network Server can come from any data server (Pandora FMS Data Server) which processes the data file, it is not necessary to assign it previously.
- **Description:** Brief description of an agent, which can include its functionality, features, location, etc.
- **Module definition|Module type:** There are two modes for an agent:
  - **Learning mode:** all the modules sent by the agent are accepted. If modules are not defined, they will be automatically defined by the system. It is recommended to activate the agents in this mode and change it once the user is familiar with Pandora FMS.
  - **Normal mode:** the modules in this mode must be configured manually. The self definition of the modules is not allowed in this mode.
- **Status:** there are two status for an agent:
  - **Disabled:** this parameter shows if the agent is not activated. Deactivated agents don't appear in the user views and don't process data nor alerts.
  - **Enabled:** this parameter is enabled if the agent is enabled and ready to receive data.

Steps to define a new agent with a network module:

1. Configure and start Pandora FMS servers.
2. Log into Pandora FMS Web Console as administrator, create an agent in Administration -> Manage Agents -> Create Agent.
3. Fill basic fields, like agents' name (very important, because this MUST be the same as reported by Pandora FMS agents). IP address (that it's used by recon server to identify existing systems and used by Network Server for default network module fields).
4. Define a existant Network Server for this agent. If there is no server assigned to this agent, it will be impossible to execute network modules for this agent.
5. If you setup this in learning mode, modules will be imported from data coming from Pandora FMS agents.
6. Install your first network module using a predefined network module (called

- Network Component), choose ICMP latency, then click on "Update".
7. Review all fields and click "Add".
  8. Start your Pandora FMS agent, and wait for the server to process data. If everything is OK, you should be able to view the data in the data view and also in the main view, after few seconds.

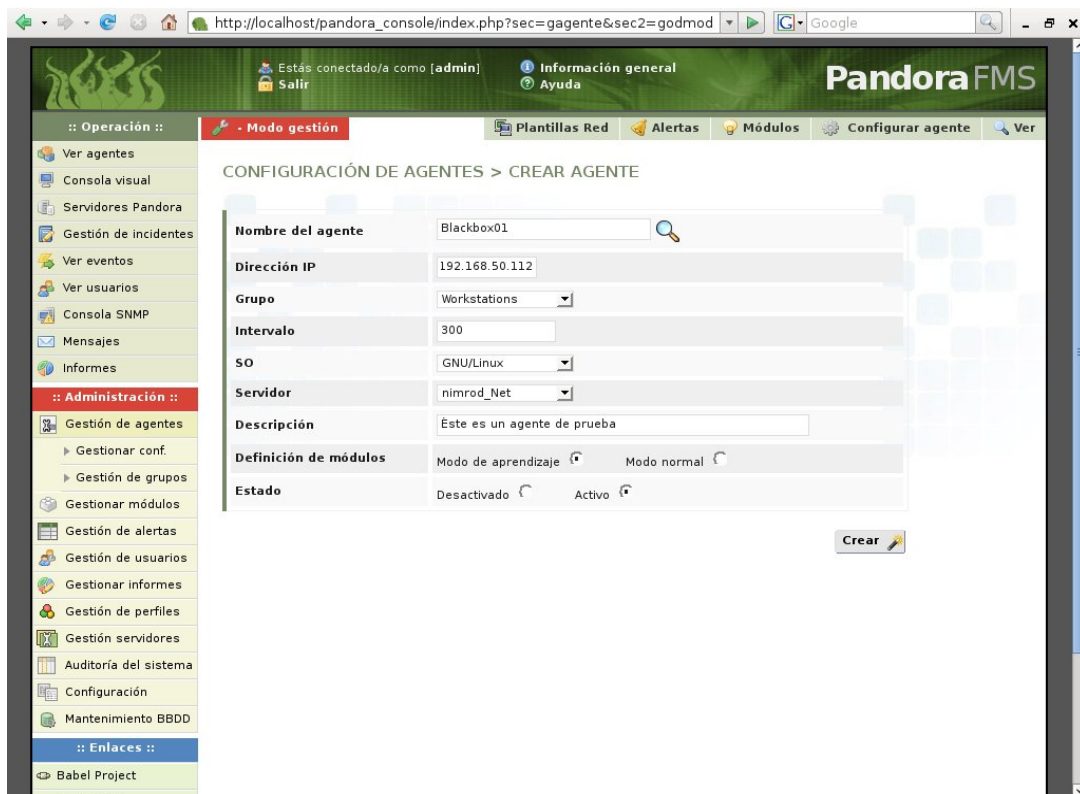


Figure 13: Create agent

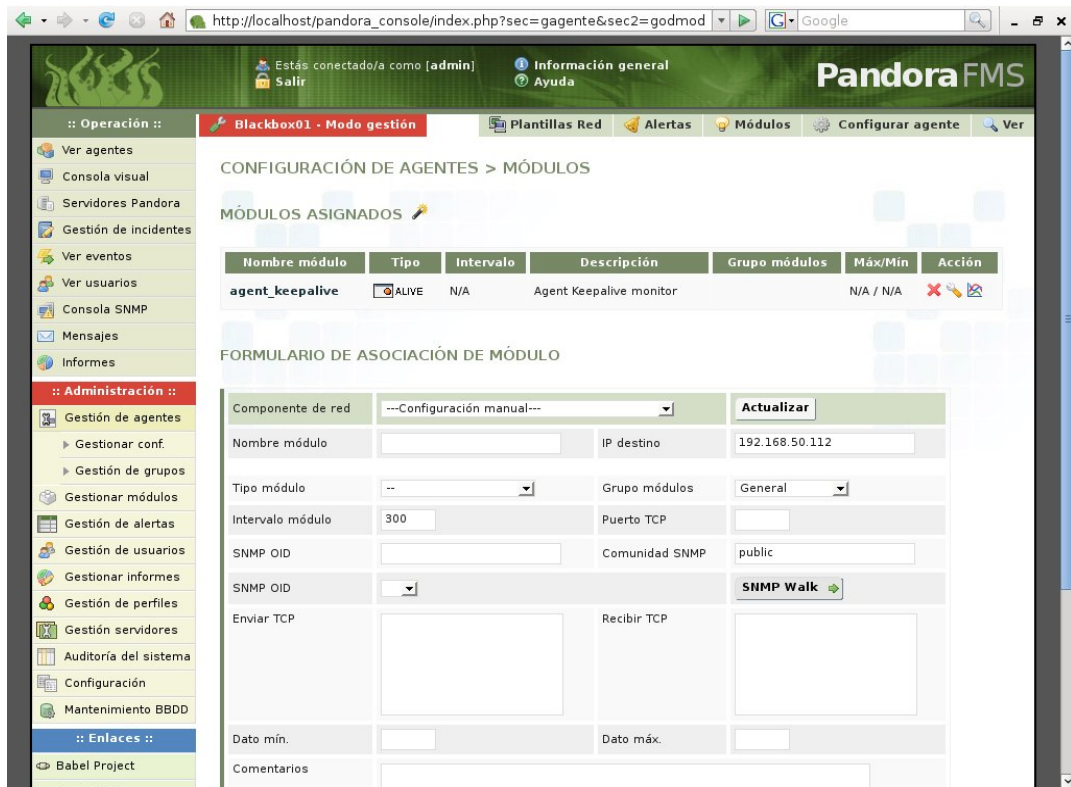


Figure 14: Fill data for agent

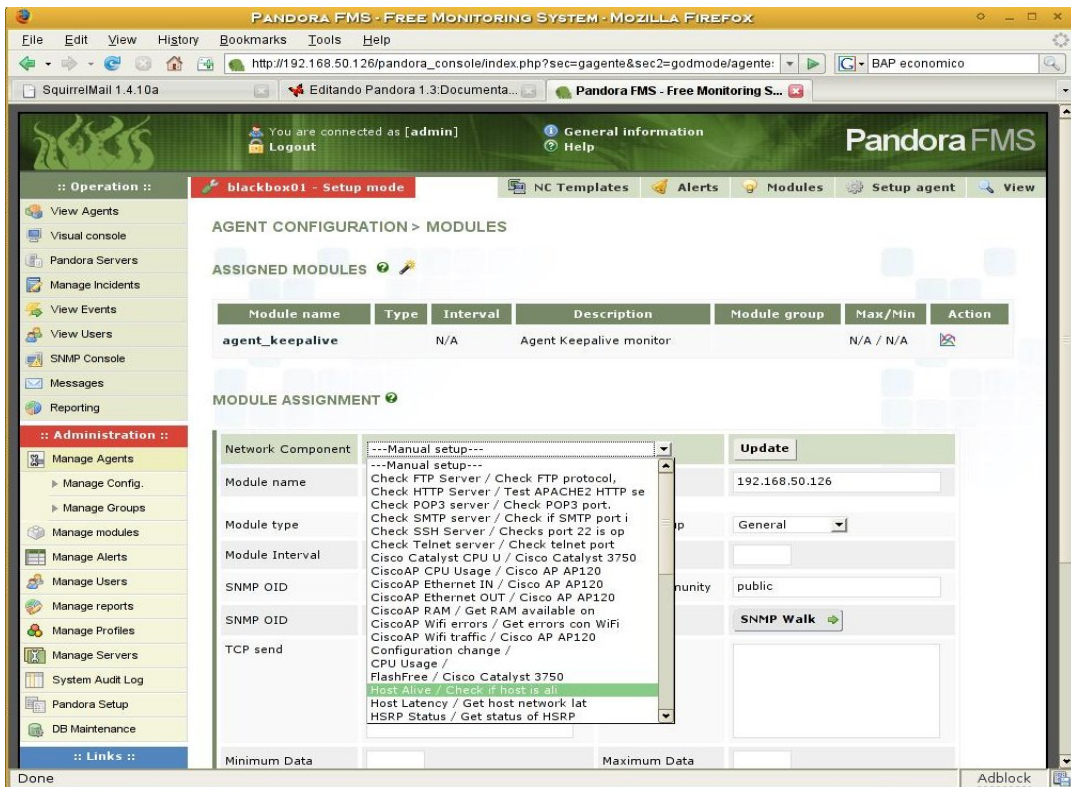


Figure 15: Create network module

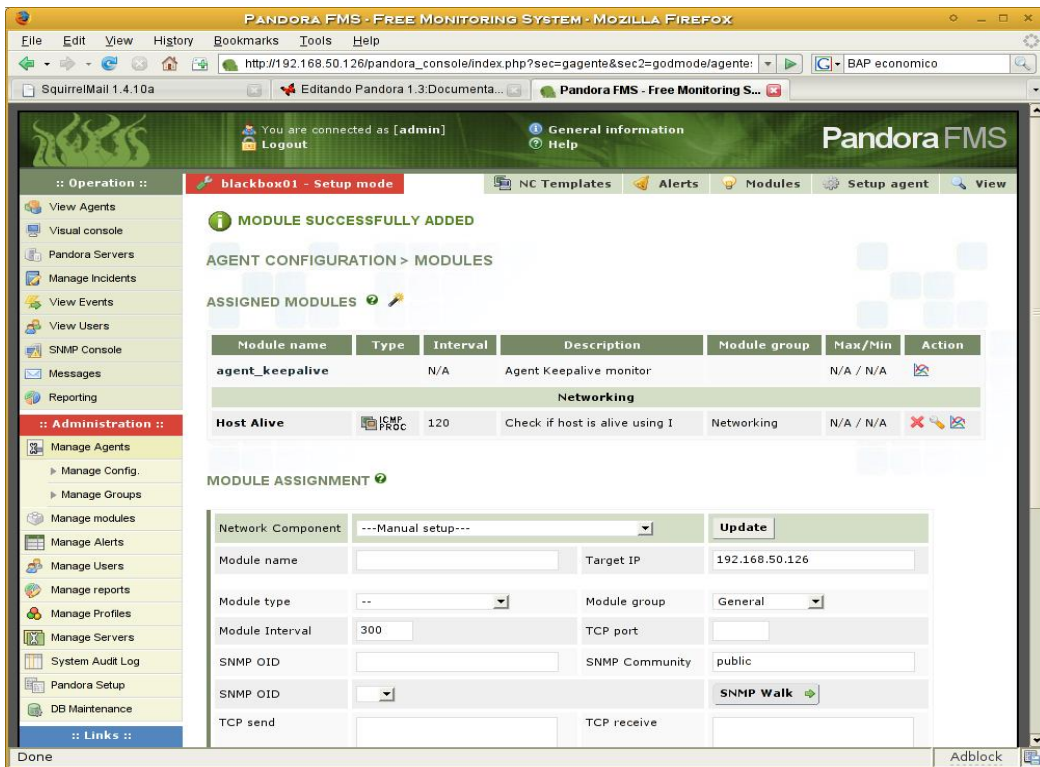


Figure 16: Adding data to module

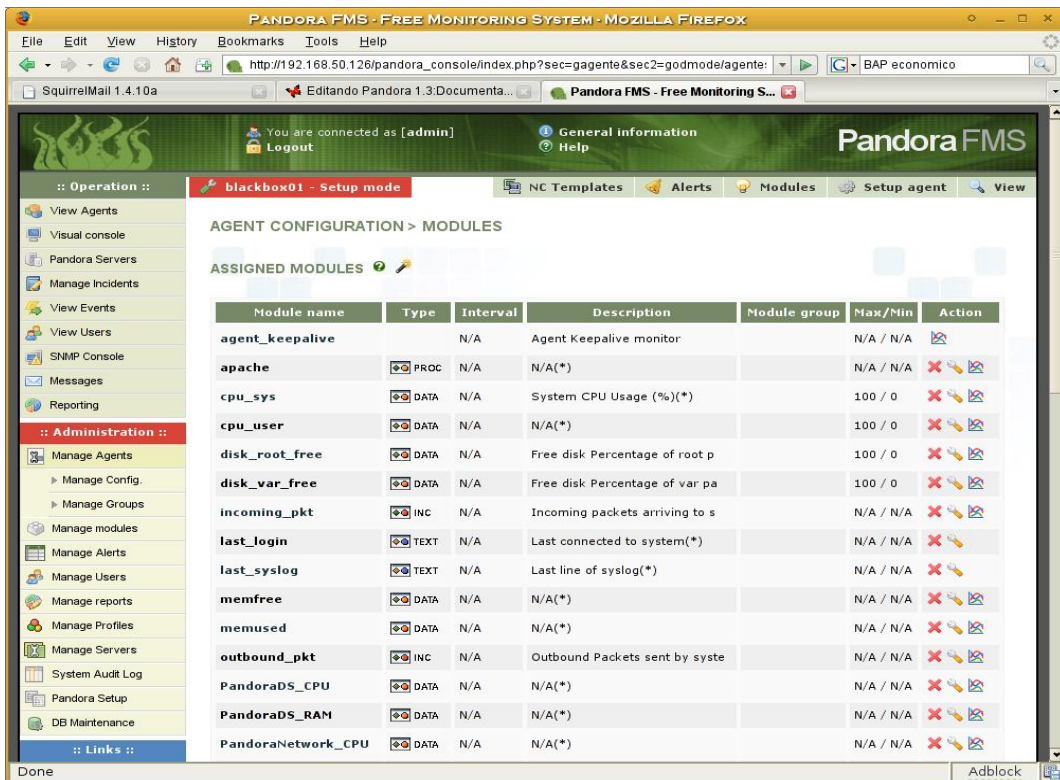
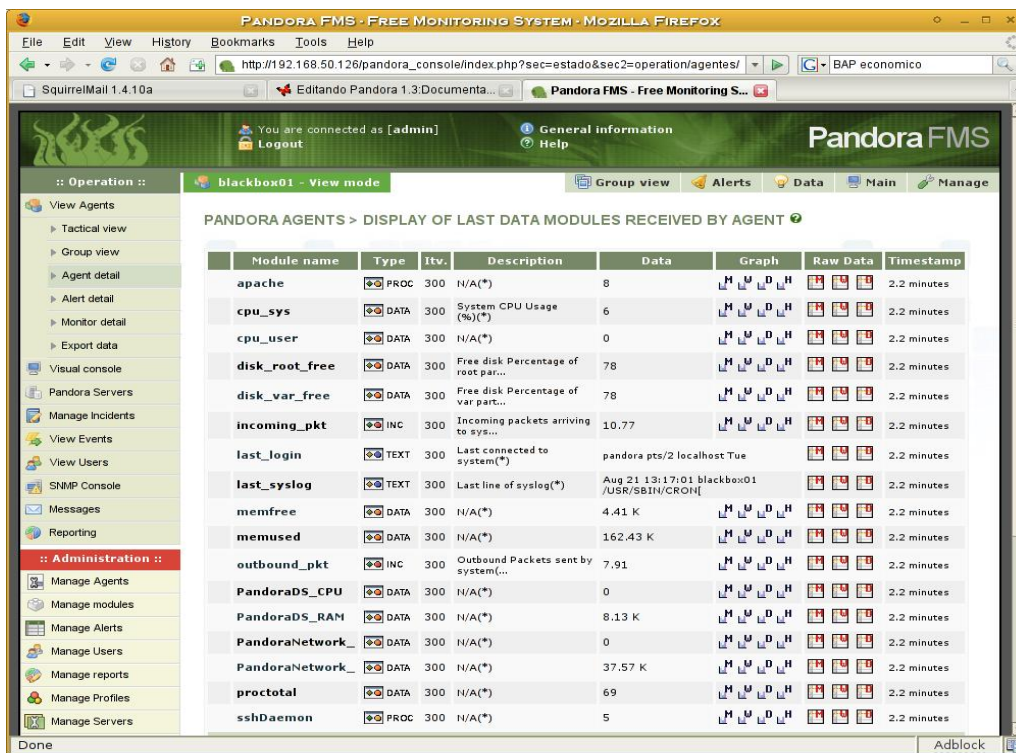


Figure 17: Review your data

The screenshot shows the Pandora FMS web interface in a Mozilla Firefox browser. The page title is "PANDORA FMS - FREE MONITORING SYSTEM - MOZILLA FIREFOX". The browser address bar shows the URL: `http://192.168.50.126/pandora_console/index.php?sec=estado&sec2=operation/agentes/`. The user is logged in as 'admin'. The main content area displays "PANDORA AGENTS > DISPLAY OF LAST DATA MODULES RECEIVED BY AGENT" for the agent "blackbox01". A table lists various system metrics with columns for Module name, Type, Itv., Description, Data, Graph, Raw Data, and Timestamp.

Module name	Type	Itv.	Description	Data	Graph	Raw Data	Timestamp
apache	PROC	300	N/A(*)	8			2.2 minutes
cpu_sys	DATA	300	System CPU Usage (%)	6			2.2 minutes
cpu_user	DATA	300	N/A(*)	0			2.2 minutes
disk_root_free	DATA	300	Free disk Percentage of root part...	78			2.2 minutes
disk_var_free	DATA	300	Free disk Percentage of var part...	78			2.2 minutes
incoming_pkt	INC	300	Incoming packets arriving to sys...	10.77			2.2 minutes
last_login	TEXT	300	Last connected to system(*)	pandora pts/2 localhost Tue			2.2 minutes
last_syslog	TEXT	300	Last line of syslog(*)	Aug 21 13:17:01 blackbox01 /USR/SBIN/CRON[			2.2 minutes
memfree	DATA	300	N/A(*)	4.41 K			2.2 minutes
memused	DATA	300	N/A(*)	162.43 K			2.2 minutes
outbound_pkt	INC	300	Outbound Packets sent by system(...	7.91			2.2 minutes
PandoraDS_CPU	DATA	300	N/A(*)	0			2.2 minutes
PandoraDS_RAM	DATA	300	N/A(*)	8.13 K			2.2 minutes
PandoraNetwork_	DATA	300	N/A(*)	0			2.2 minutes
PandoraNetwork_	DATA	300	N/A(*)	37.57 K			2.2 minutes
proctotal	DATA	300	N/A(*)	69			2.2 minutes
sshDaemon	PROC	300	N/A(*)	5			2.2 minutes

Figure 18: Review your data



The screenshot shows the Pandora FMS web interface in Mozilla Firefox. The user is logged in as 'admin'. The main content area displays 'PANDORA AGENTS > DISPLAY OF LAST DATA MODULES RECEIVED BY AGENT' for agent 'blackbox01'. A table lists various system metrics and their values.

Module name	Type	Itv.	Description	Data	Graph	Raw Data	Timestamp
apache	PROC	300	N/A(*)	8	[Graph Icon]	[Raw Data Icon]	2.2 minutes
cpu_sys	DATA	300	System CPU Usage (%)	6	[Graph Icon]	[Raw Data Icon]	2.2 minutes
cpu_user	DATA	300	N/A(*)	0	[Graph Icon]	[Raw Data Icon]	2.2 minutes
disk_root_free	DATA	300	Free disk Percentage of root par...	78	[Graph Icon]	[Raw Data Icon]	2.2 minutes
disk_var_free	DATA	300	Free disk Percentage of var part...	78	[Graph Icon]	[Raw Data Icon]	2.2 minutes
incoming_pkt	INC	300	Incoming packets arriving to sys...	10.77	[Graph Icon]	[Raw Data Icon]	2.2 minutes
last_login	TEXT	300	Last connected to system(*)	pandora pts/2 localhost Tue	[Graph Icon]	[Raw Data Icon]	2.2 minutes
last_syslog	TEXT	300	Last line of syslog(*)	Aug 21 13:17:01 blackbox01 /USR/SBIN/CRON[	[Graph Icon]	[Raw Data Icon]	2.2 minutes
memfree	DATA	300	N/A(*)	4.41 K	[Graph Icon]	[Raw Data Icon]	2.2 minutes
memused	DATA	300	N/A(*)	162.43 K	[Graph Icon]	[Raw Data Icon]	2.2 minutes
outbound_pkt	INC	300	Outbound Packets sent by system(...	7.91	[Graph Icon]	[Raw Data Icon]	2.2 minutes
PandoraDS_CPU	DATA	300	N/A(*)	0	[Graph Icon]	[Raw Data Icon]	2.2 minutes
PandoraDS_RAM	DATA	300	N/A(*)	8.13 K	[Graph Icon]	[Raw Data Icon]	2.2 minutes
PandoraNetwork_	DATA	300	N/A(*)	0	[Graph Icon]	[Raw Data Icon]	2.2 minutes
PandoraNetwork_	DATA	300	N/A(*)	37.57 K	[Graph Icon]	[Raw Data Icon]	2.2 minutes
proctotal	DATA	300	N/A(*)	69	[Graph Icon]	[Raw Data Icon]	2.2 minutes
sshDaemon	PROC	300	N/A(*)	5	[Graph Icon]	[Raw Data Icon]	2.2 minutes

Figure 19: Review your data

## 5.5 VIEWING DATA WITH PANDORA FMS

Pandora FMS main function is to show the status of every monitored element. For that, there are several views in the "View agents" menu.

### 5.5.1 Tactical view

The tactical view allows you to see the main status of the whole system and every monitored element with a simple view. The elements mentioned are:

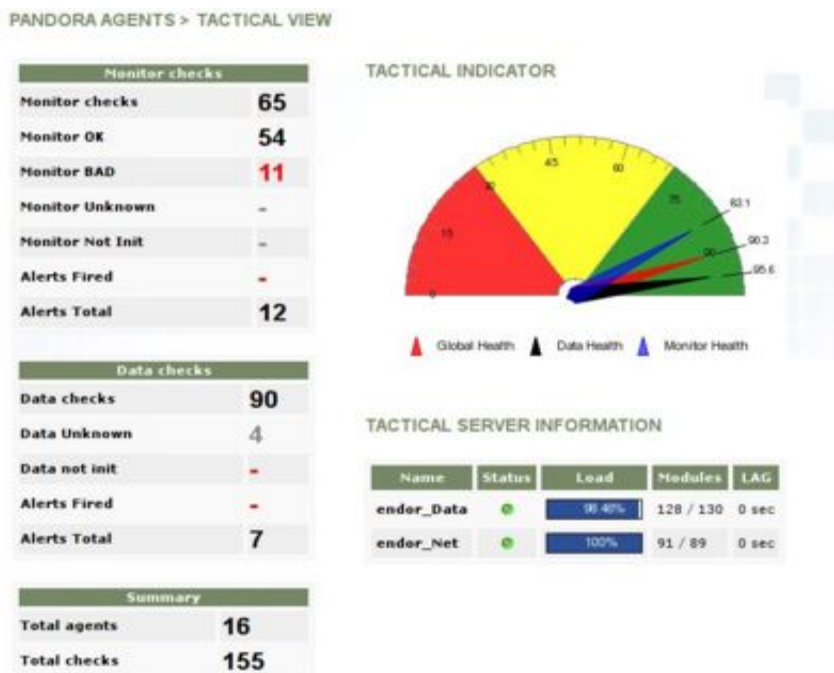


Figure 20: Main tactical view of Pandora FMS

- **Monitors:** A monitor is a "Proc" module type, this means that can tell us directly if something is OK or not, without any extra processing or any alert. The status of a monitor can be fine (**OK**), wrong (**BAD**) or unknown (**Unknown**). A element is in a Unknown status if it didn't report any data for a time double than its interval. The Unknown status doesn't mean that there is something wrong, it just means that, no data from that element. In some specific modules, such as *remote\_snmp\_proc*, you can set that the "Unknown" status can be considered as "BAD". Other modules, like *remote\_icmp\_proc* can only return "OK" or "BAD", since "Unknown" has no sense. A "Not initialized" status means that, for any reason, usually a bad module or server configuration, it was never initialized, this is, it reports a "Unknown" status but it status was never "OK" nor "BAD". Those "Not initialized" modules are removed from the configuration after running the Database maintenance script.
- **Data and alerts.** Like monitors process information, data work with not quantifiable pieces of information. So that there are alerts, to set which values are correct and which one are not.

It is also mention the total number of agents and checks. For Pandora FMS a check is every information processed, this is, modules.

In this view there is also a short information column about the status of the Data and Network Servers. The servers show the modules they handle, and the total number of modules that they can handle. It is also shown the delay or lag while showing the modules.

### 5.5.2 Group view

The group view allows to see at first view all the information monitored and managed by Pandora FMS disaggregated by groups. Groups are logical items that group Pandora FMS logical agents. They can have any name and icon. Each group icon shows a lot of



information.

In this view you can see the general status of the group. A green bubble shows that there are modules with a "OK" status. A red bubble shows that there is, at least, a module with a "BAD" status. A grey bubble shows that there are monitors with "Unknown" status. A yellow bubble shows that recently some alert was fired. The groups frame shows the same but following some priority: yellow -> red -> grey -> green, assuming then that if an alert was fired, the frame will be yellow, if not but there is a module in a wrong status, then it will be red, and so on.



Figure 21: View of a group

The green button of the top right corner, is to force the immediate execution of all network modules, regardless of their refresh time.

This view shows the detail of each "bubble", stating the number of modules and agents the group contains. Is necessary to move the mouse pointer over it to see this view.

### 5.5.3 Agent global detail

In this view you can see almost every global detail of any Pandora FMS logical agent. Sorted, you can analyze: agent's name, Operating System, interval, agent's group and a group classification: total modules, monitor modules (if there are modules in bad status they will appear in red, and modules in unknown status will appear in grey). The general status of the agent is based on whether it has monitors, and if any of those is in a bad status, a red frame will appear. If every module is OK, then the frame will be green. If any alert was fired, then the frame will be yellow, and if it has no monitor agents but data, then it will be blue.



Figure 22: Detailed view of a group

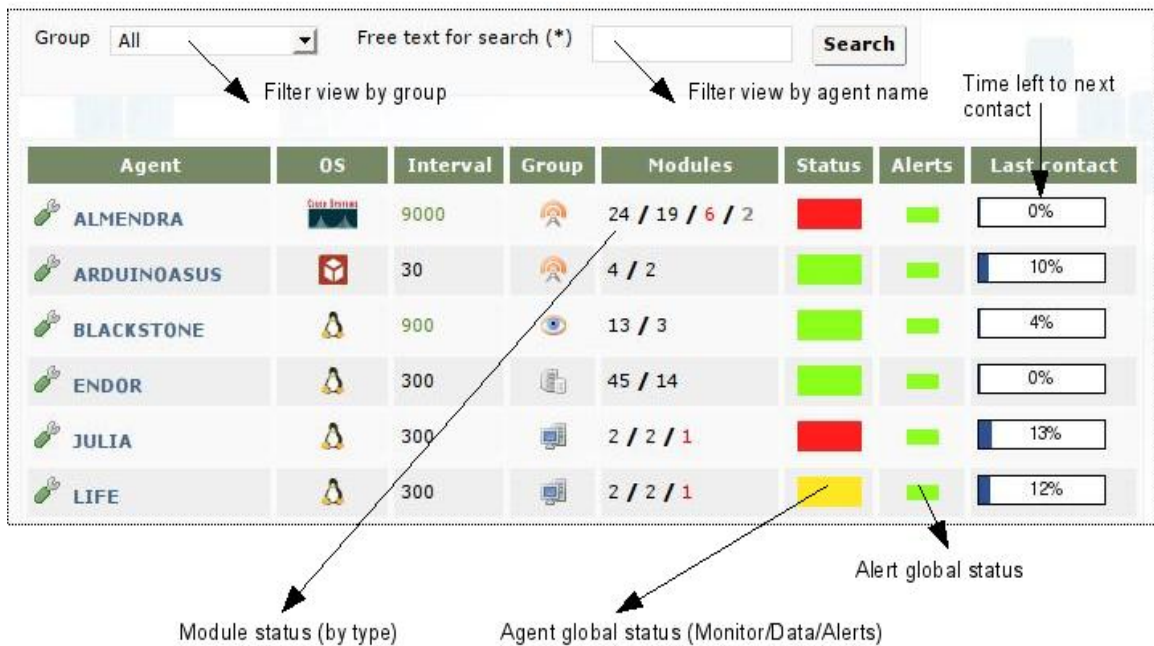


Figure 23: Agent detail screen explained

There is also a smaller box, next to the status one, that shows the detail of the alerts. If any alert was fired recently (recently refers to *time threshold*), then the box color will be red. If there was not any fired alarm, then the box color will be green.

Finally, there is a progress bar at the right (placing the mouse over it you will be shown the date of the last contact) that shows, approximately, the time left until the next contact with the agent. Out of that time, the agent will be considered as "down", since there is no contact with any of its data sources. An agent could be active, but any of its modules could be not sending information (unknown status). This can be seen in the detailed view of the agent, as you can see ahead.

This view can be paged and filtered by groups to show all the information monitored by Pandora FMS. This view is autorefreshed every five minutes.

### 5.5.4 Details of Pandora FMS logic agent

The view of Pandora FMS logic agent is the main part of Pandora FMS monitorization. From here you get all the information from the agent. Its data, its status, history, graphics of every element, numeric and string data historic tables, and graphical statistics of the frequency of the connection.



Figure 24: Details of an agent

Agent's view allows to browse through different options by tabs placed on top right. Agent's view also allows us to go to configuration mode using the tabs system.

The main view shows us all the general info of the agent, as well as the total packets it has, their distribution (by module), the agent's data contact frequency, and other general data, like the last remote and local contact date, and the Network Server where the agent is assigned for the execution of its tasks.

### 5.5.5 Monitor detail

In Pandora FMS monitors show, in a easy way, the status of any monitorized element, which can be whether a process is running or not, or if some interface is working or not.

Therefore everything is related with a correct status (green) or an incorrect one (red). In the main view of Pandora FMS agent you can see, just below the general info, the status of the associated monitors to that logical agent.

**FULL LIST OF MONITORS**

Type	Module name	Description	Status	Interval	Last contact
TCP PROC	Check SSH Server	Checks port 22 is opened		--	4 minutes
PROC	dhcp server	N/A(*)		--	2.9 minutes
PROC	DNS_CHECK	Check that dynamic DNS name corresp		--	2.9 minutes
ICMP PROC	Host Alive	Check if host is alive using ICMP p		120	1.3 minutes

Figure 25: Monitors details

**5.5.6 Alert detail**

**FULL LIST OF ALERTS**

Type	Name	Description	Min.	Max.	Time threshold	Last fired	Times Fired	Status	Validate
eMail	ARP Destroy	Very high rate of ARP destroyed entries	0	0.50	5 minutes	1.1 months	0		<input type="checkbox"/>
eMail	DNS_CHECK	DNS for artica.homelinux.com is not sync	1	1	30 minutes	2.3 hours	0		<input type="checkbox"/>
eMail	Host Alive	System down	1	1	Two hours	+6 months	0		<input type="checkbox"/>
Internal Audit	MailServer	Sendmail is down	1	3	5 hours	5.8 months	0		<input type="checkbox"/>

Figure 26: Alerts details

This view, which appears at the bottom of the detailed view of the agent, shows the status of every alert defined for this agent. Its type, name, description, trigger conditions (min/max allowed values before triggering), alert time to live (or time interval in which the alert is checked since first failure), last time it was fired, and number of times it has been triggered in this interval (time to live). In Pandora FMS v.1.3.1 there is a new validation button to mark the triggered alerts as "validated" and remove their red status.

This view also appears just below the monitors list, in the main view of the agent.

**5.5.7 Data detail**

Pandora FMS stores all the data from every monitor for long time (defined by the user). Pandora FMS is able to represent graphically all that data, and show a table with the history of them.

This view shows the last value of each module and its date of reception, and allows to show its representation in a table or a graphic. To get the graphic you have to ckick on the small graphic icon. You can choose to show a montly, weekly, daily or hourly scale for any

data (except string). You can also choose to show that data in a table, clicking on the small icons at the right side of the data row of the module.

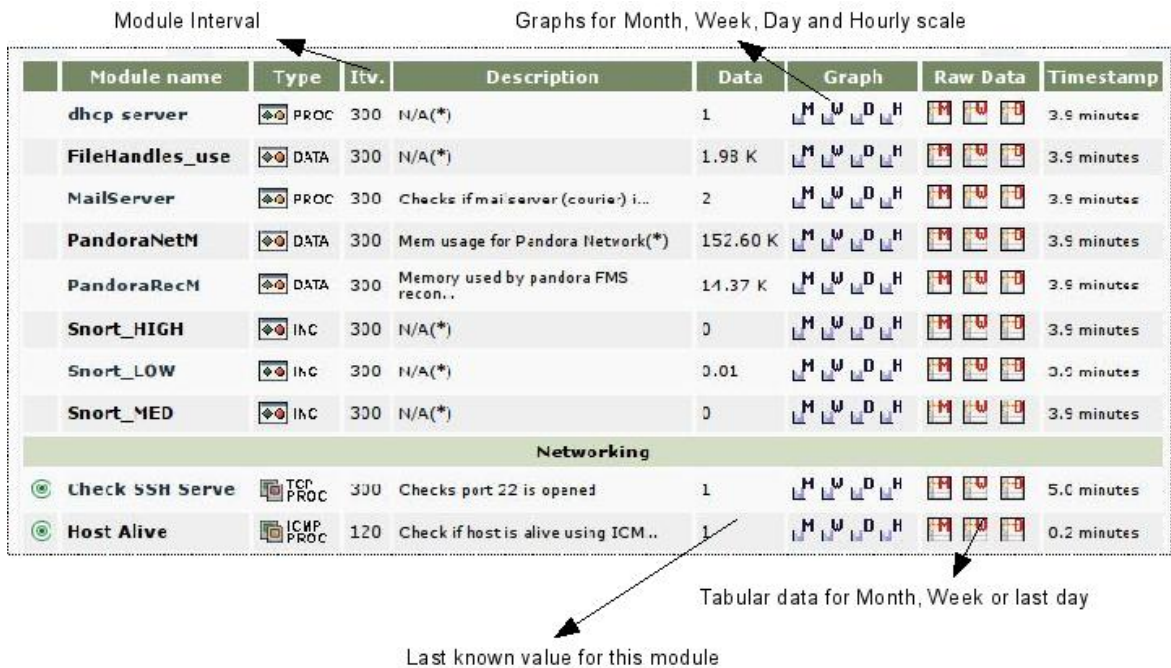


Figure 27: Detail screen explained

Pandora FMS plots can be modified, and represent multiple plots. They show real time data, which means it is generated every time the user needs one, and they show the actual status of that monitor.

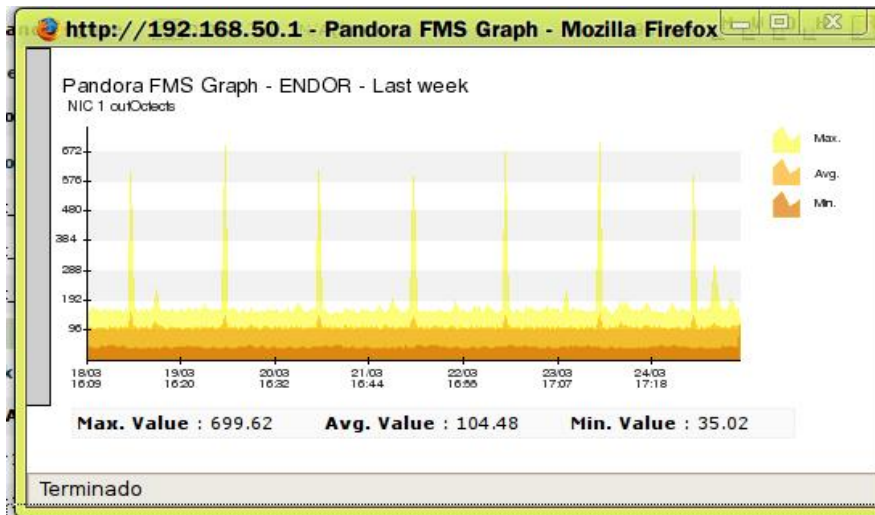


Figure 28: Pandora FMS activity graph

There is a small menu that allows (placing the mouse over the tab at the left of the graph window) reconfiguring the plot to show it bigger, with a longer time scale, to show, on top of the graphic, the defined alerts that module has, or to plot with red bars every event

happened in the scope of the agent of that module.

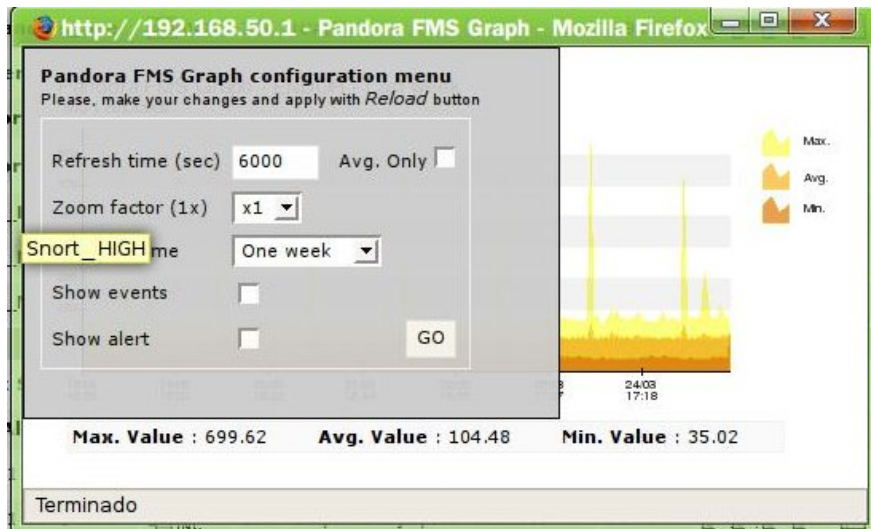


Figure 29: Pandora FMS graph configuration menu

### 5.5.8 Exporting data

Pandora FMS allows to export the data of any agent module to several output formats, using a simple Web tool. This tool allows to export in .CSV (*comma separated values* file) that can be easily processed by spreadsheets such as Excel or OpenCalc. It also allows to show data in tables, like in the agent data view, or show a summarized table of average values per day of the week.

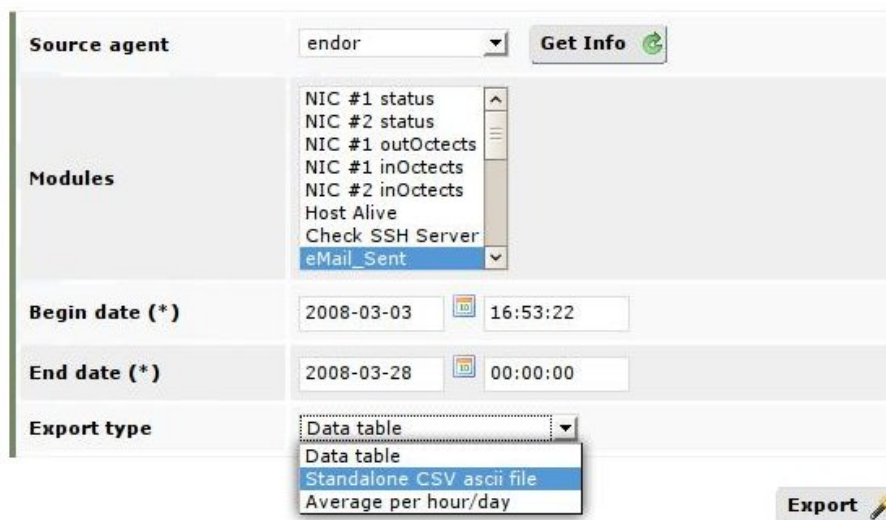


Figure 30: Pandora FMS exporting screen

If you need to export a whole agent (or even several), the code of this section is very easy and can be used to dump data to any required format with little modifications.

## 5.6 BASIC INCIDENT MANAGEMEN

---

The process of monitoring systems, as well as receiving and processing data in order to monitor them in the specific defined time, it is necessary to track any incident that can happen in those systems.

For that, there is an incident manager where every user can open incidents, detailing what happened in the network, and update them with comments and files every time there is any news.

This system allows team work, with different roles and workflow systems that allows an incident to go from a group to another, and that members of different groups, and different users, work together on the same incident, sharing information and files.

Going to "Manage incidents", from "Operation" menu, you can see a list with all the incidents, sorted by date of update. Using filters you can see only the incidents you are interested in.

You can combine filters, filtering then by:

- Incident status. Where you can see:
  - All the incidents
  - Active incidents
  - Closed incidents
  - Rejected incidents
  - Expired incidents
- Incident priority. Where you can see:
  - All the priorities
  - Informative
  - Low
  - Medium
  - Serious
  - Very Serious
  - Maintenance
- Filtering by groups. Where you can see incidents attached to every existing group in Pandora FMS.

At the incident list, each one appears with information distributed as follows:

**ID:** Incident identifier.

**Status:** Actual status of the incident, with with following icons:

- Active incident
- Active incidents, with comments
- Rejected incidents
- Closed incidents
- Expired incidents

**Incident name:** Incident assigned name.

**Priority:** Assigned priority to the incident, with colored icons.

**Group:** Defines the group where where the incident is attached. An incident can only be attached to one group.

**Updated at:** Last time the incident was updated.

**Source:** This tag is used to assign the incident a source. It can be selected form a list stored in the Database. Although the sources list is fiexd and predefined, the Database administrator can modify it.

**Owner:** The user who has the incident assigned. Not to be confused with the incident creator, since the owner of it can have changed. The owner can assign the incident to another user, as well as any user with incident management privileges over the group where the incident belongs.

### 5.6.1 Adding an incident

To create an incident just go to Manage incidents -> Create incident, from "Operation" menu.

You must fill every field, once done, click on "Create".

### 5.6.2 Tracking an incident

From Operation -> Manage incidents you can track every incident.

Choose an incident from the column "Incident" and click over it. You will see all the configuration data of the incident and also the attached notes and files.

You can update the fields: Incident, Owner, Status, Source, Group, Priority, and description from the shown page.

### 5.6.3 Adding notes to an event

To add notes to an existing incident, click on "Add note", and you will see a page with a text area. Write the note and click "Add" when finished.

Attached notes to incidents can be seen at the description page of the incident, below the report of the incident.

Any user with reading rights over an incident can add a note. Only the owners of the incident or the note, can delete them.

### 5.6.4 Adding files to an event

Sometimes is interesting to attach an image, a config file, or any kind of file to an incident.

To do so, go to "Add file", clicking on the diskette icon and two entry fields will be shown. Searth the file in your localhost and then write a description (optional). When you have finished, click "Upload" to start uploading the file to the server.



Any user with reading rights over an incident, can add a file. Only the owners of the incident or the file, can delete them.

To track the incident, you can access to every attached file clicking on "Attached files" in the same window.

### 5.6.5 Creating an incident from an event

Now with Pandora FMS you can create incidents from events, just go to the section "View events" from "Operation" menu, and you will find an icon below the column "Action", of the table that shows the events. See the screenshot below:

Status	Event name	Agent name	Group	User ID	Timestamp	Action	All
	Monitor (Host Alive) goes down	SILVIOJOSE			2008-03-24 18:17:43		<input type="checkbox"/>
	Monitor (Check ssh Server) goes down	LIFE			2008-03-24 17:44:24		<input type="checkbox"/>
	Monitor (Host Alive) goes down	LIFE			2008-03-24 17:41:49		<input type="checkbox"/>

Figure 31: Event screen with the icon highlighted

**INCIDENT MANAGEMENT > CREATE INCIDENT**

**Incident**

**Opened at** 2008/03/24 19:13:04 **Updated at** 2008/03/24 19:13:04

**Owner**  **Status**

**Source**  **Group**

**Priority**  **Creator** admin ( Default Admin )

**Create**

Figure 32: Creating an incident from an event

### **5.6.6 Autogenerated incidents (Recon Server)**

With Recon Server integration, also the auto-generation of incidents from the events processed by the Recon Server has been added. These events include new host detection. These events are exactly the same as the rest, and they also appear listed at "Manage incidents" section of "Operation" menu.

### **5.6.7 Searching incidents**

If you want to search an incident from all the incidents created inside Pandora FMS, besides the filters seeing before, you can perform a fine tuned search from Manage incidents -> Search incident, inside the "Operation" menu.

From here you can search any text string inserted as substring along with the incident, filtering by the user who created the incident. Searches are performed over the incident's title, or over the content of it, but not over the attached notes or files. You can also combine these searches with the group, priority or status filters.

### **5.6.8 Statistics**

From Manage incidents -> Statistics, inside the "Operation" menu, you have access to five different types of incident statistical graphs:

- Incidents by status
- Incidents by priority
- Incidents by group
- Incidents by user
- Incidents by source

## **5.7 USERS IN PANDORA FMS**

---

A user is defined by its daily activity as user. Each user has his/her own profile, with a list of actions that can or cannot perform, to access to Pandora FMS. One or more profiles can be assigned to a single user. Each user is given a number of groups of agents he/she has permission to access, as well as the administrative profile he/she will have in each group. Each user can belong to one or more groups, with an assigned profile for each of them.

The agent belongs only to one group, sharing the group with agents of similar characteristics. Groups also contain incidents.

Summarizing: User profiles in Pandora define which users can access Pandora FMS as well as what can each user do. Groups define elements in common among several users. Each user can be in one or more groups at the same time. Each group has user profiles defined and attached to it. A profile is a list of things that a user can do, such as view incidents, manage the database or other tasks.

### 5.7.1 Profiles

Pandora profile manager is used to assign specific profiles to each user. To use the profile manager, go to Administration -> Manage profiles. A hierarchy of users is created, structured by the user profile within the company. With this system different security levels can be implemented: read-only users, agent group coordinators or system administrators.

Any of the following roles can be assigned to a new profile:

- View incidents (**IR**).
- Edit incidents (**IW**).
- Manage incidents (**IM**).
- View agents (**AR**). To view agents as well as the events generated by them.
- Edit agents (**AW**). To modify then agent's modules.
- Edit alerts (**LW**). To modify the alerts assigned to an agent.
- Manage users (**UM**). To modify users and their roles.
- Manage DB (**DM**). To modify the configuration and data of the database (global).
- Manage alerts (**LM**). To define new alerts (global).
- Manage Pandora FMS (**PM**). To modify general system settings.

## 6 ADVANCED MONITORING

---

### 6.1 ALERTS IN PANDORA FMS

---

An alert is Pandora FMS reaction to an "out of range" module value. The alert can consist of sending an e-mail or an SMS to the administrator, sending a SNMP trap, write the incident into the system log or into Pandora FMS log file, etc. Basically, an alert can be anything that can be triggered by a script configured in the Operating System where Pandora FMS Servers run.

Alerts can be disabled individually or disabling an whole group of agents. Also if you disable an agent, it would not trigger any alert.

Pandora FMS 1.3.x does not allow *alert linking* following a logic sequence. However it is planned for Pandora 2.x versions.

#### 6.1.1 Adding and editing alerts

The existing alerts are accessed clicking on the "Manage alerts" option, of the Administration menu. In that section you can modify or add your own alerts. The server (Data or Network Server) executes the alerts, processing the agent module. Alerts are also executed with privileges of the user that runs Pandora FMS server.

There are some predefined alerts, in which is very likely you will have to adjust, in case your system does not provide the internal commands needed to execute those alerts. The development team has tested these alerts with Red Hat Enterprise Linux (RHEL), CentOS, Debian and Ubuntu Server.

- **eMail.** Sends an e-mail from Pandora FMS Server. It uses your local *sendmail*. If you installed other kind of local mailer or do not have one, you should install and configure sendmail or any equivalent (and check the syntax) to be able to use this service. Pandora FMS rely on system tools to execute almost every alert, it will be necessary to check that those commands work properly on your system.
- **Internal audit.** This is the only "internal" alert, it writes the incident in Pandora FMS internal audit system. This is stored in Pandora FMS Database and can be reviewed with Pandora FMS audit viewer from the Web console.
- **LogFile.** Saves information about the alert inside a text file (*.log*). Use this type of alert to generate log files using the format you need. To do so, you will need to modify the command so that it will use the format and file you want. Note that Pandora FMS does not handle file rotation, and that Pandora FMS Server process that executes the alert will need access to the log file to write on it.
- **SMS Text.** Sends an SMS to a given cell phone, of course, you need to define a alert before making this possible, and you also need a configured gateway,

accessible from Pandora FSM Server. You can also setup Pandora FMS using **Gnokii** to send it directly using a Nokia cell with a USB cable, see the documentation below to learn more about this.

- **SNMP Trap.** Sends a SNMP Trap.
- **Syslog.** Sends an alert to the system log, it uses the system command "*Logger*".

The values "\_field1\_", "\_field2\_" and "\_field3\_" of the customized alerts are used to build the command line that will be executed.

When a new alert is created the following fields must be filled in:

- **Alert name:** The name of the alert. It is important to describe correctly its function, but briefly, for example: "Comm. log".
- **Command:** Command that the alert will trigger, tis he most important field while defining an alert. Note that the macros *\_field1*, *\_field2*, and *\_field3* are used to replace the configured parameters at the alert definition. That way the execution of the command fired by the alert is built. While defining an alert, you should test the correct execution of the alert, and that the result is the expected (send an email, generate an entry in a log, etc) at the command line.
- **Description:** Long description of the alert, optional.

The complete set of macros that can used within an alert is the following:

- *\_field1\_*: Usually used as username, phone number, file to send or e-mail destination.
- *\_field2\_*: Usually used as short description of events, or subject line for e-mails.
- *\_field3\_*: A full text explanation for the event, can be used as the text field for an email or SMS.
- *\_agent\_*: Full agent name.
- *\_timestamp\_*: A standard representation of date and time. Automatically replaced when the alert is executed.
- *\_data\_*: The data value that triggered the alert

## 6.1.2 Alert definition examples

### *Adding a new alert: Sending emails with expect*

Sometimes we need to use a authenticated SMTP to send mails. It is probably easier to use a simple *EXPECT* script instead configuring *sendmail* to use an authenticated SMTP. This is an example using *EXPECT* to send mails using a *Exchange* server:

```
#!/usr/bin/expect -f
set arg1 [lindex $argv 0]
set arg2 [lindex $argv 1]
set arg3 [lindex $argv 2]
set timeout 1
spawn telnet myserver.com 25
```

```

expect "220"
send "ehlo mymachine.mydomain.com\r"
expect "250"
send "AUTH login\r"
expect "334"
send "2342348werhkwjernsdf78sdf3w4rwe32wer=\r"
expect "334"
send "YRejewrhneruT==\r"
expect "235"
send "MAIL FROM: myuser@domain.com\r"
expect "Sender OK"
send "RCPT TO: $arg1\r"
expect "250"
send "data\r"
expect "354"
send "Subject: $arg2\r"
send "$arg3 \r\r"
send ".\r"
expect "delivery"
send "quit"
quit

```

#### Set permissions for */root/smtp*:

```
chmod 700 /root/smtp
```

before using it, and of course, make sure that */usr/bin/expect* is working fine.

To use this with Pandora FMS, you need to create a new alert (or modify the email alert existing one) and specify the following fields at the Pandora FMS alert definition:

```
/root/smtp _field1_ _field2_ _field3_
```

Of course the script can be placed anywhere. You just need to be sure that the server that processes the data launches the script alert: if it is a network data, then it will be the Network Server, if it is data from an agent, through a data XML file, then it will be the Data Server. If you have different physical servers, you may need to copy the same script to the same location, with the same permissions and the same owner to every system where you have a Pandora FMS Server that you want to execute that alert. Also note that Pandora FMS Network Servers need to be run as *root* (so they can perform ICMP latency tests) and the Data Servers can be run as a normal user, without privileges. The alert will be executed by the user running Pandora FMS server process.

#### Adding a new alert: Sending SMS with *Gnokii*

In order to use Gnokii you need a compatible cell phone, Nokia or other (check it at [Gnokii projec web page](#)). You will also need a USB data cable attached to the cell phone and to the Pandora FMS Server you want to use to send the messages. Gnokii supports a vast number of Nokia phones (and some other brands). With Gnokii you can send SMS from the command line. This way is very easy and fast sending SMS directly from a Pandora FMS server, avoiding the usage of SMS gateways or costly hardware systems. Another alternative to Gnokii is the [Gammu project](#).

### SMS send from the command line example with Gnokii:

```
echo "PANDORA: Server XXXX is down at XXXXX" | gnokii --sendsms  
555123123
```

Gnokii cannot send MMS with images attached, but you can add a HTTP/WAP URL with image you want to send, for example:

```
echo "Image capture sample" | gnokii --sendsms 555123123 -w  
http://artica.homelinux.com/capture.jpg
```

You can send the image URL, or an URL that takes you to a Pandora FMS light weight console, so you can see the console from the mobile device and check the data.

The developpe team has tested SMS sending process from a Nokia 6030, sending SMS alerts when the Internet access was down. The Nokia 6030 cell phone uses the module definition 6510 available at the file *gnokiirc* and takes around four seconds to send an SMS.

You can implement a more powerful SMS gateway using *Gammu*. At the end of the documentation you can learn how to implement a SMS sending server with a queue system that allows to implement a SMS shipment network server.

### 6.1.3 Assigning Alerts to modules

The next step after adding an agent, having configured its modules, and defined the alerts, is assigning those alerts to the agent. This step is necessary to establish alert conditions in those desired cases. This is done by clicking on the agent to be configured in the "Manage agents" option, from Administration menu, or using the edition mode and selecting the tab "Alerts", from the agent view.

Alert type	eMail	Alert status	Enabled
Min. Value		Max. Value	
Alert text			
Description			
Field #1 (Alias, name)			
Field #2 (Single Line)			
Field #3 (Full Text)			
Time from	00:00	Time to	00:00
Time threshold	5 Min.	Other	
Min. number of alerts	0	Max. number of alerts	1
Assigned module	ARP Alloc ( generic_data_inc )		

Figure 33: Adding new alert to a module

The next fields must be filled to assign an alert:

- Alert type: This can be selected from the alert list previously generated.
- Max. Value: Defines the maximum value for a module. Any value above that threshold will trigger the alert.
- Min. Value: Defines the minimum value for a module. Any value below that will trigger the alert. "max." & "min." couple are the key values while defining an alert, since they define the range of normal values, out of that range Pandora FMS will trigger the alert.
- Alert text: In case of string modules, you can define a regular expression or a single string to match contents of data module to trigger the alert.
- Time from / Time to: This defines a range of "valid" time range to fire alerts.
- Description: Describes the function of the alert, and it is useful to identify the alert among the others in the general view of alerts.
- Field #1 (Alias, name): Define the used value for the "\_field1\_" variable.
- Field #2 (Single Line): Define the used value for the "\_field2\_" variable.
- Field #3 (Full Text): Define the used value for the "\_field3\_" variable.
- Time threshold: Time counter since the first alarm was triggered (or condition to trigger it) . During that time, the alerts are handled with the rest of the parameters (Min. number of alerts, Max. number of alerts). You can choose between the interval configured or define other interval.
- Min. number of alerts: Minimum number of alerts needed to start triggering an alert. Works as a filter, needed to remove false positives.
- Max. number of alerts: Maximum number of alerts that can be sent consecutively



during the same time threshold.

- Assigned module: Module to be monitored by the alert.

All the alerts of an agent can be seen using the "Alerts" tab. Let's see an example:

*"I want to fire an alert when XXX goes down, and please, don't disturb me again at least for one hour. After that time, if it is still down, fire another alert and wait another hour".*

You need to setup:

- Time threshold 3600 (1 hour).
- Min. number of alerts = 1.
- Max. number of alerts = 1.

#### 6.1.4 Pandora FMS alert log

Every triggered alert is registered in Pandora FMS event viewer. By default, Pandora FMS defines an output to a plain text file, as an alert. This alert can be used to integrate Pandora FMS with other software (Babel, OSSIM, etc.).

Pandora FMS alert log is defined as follows:

```
<yyyy-mm-dd hh:mm:ss> pandora <agent_name> <data> <field1> <field2>
```

For example:

```
2008-03-31 13:43:01 pandora agent01 134 google_latency Too high google
latency
```

Fields *field1* and *field2* are user-defined, at the alert parameters. They should be a short name and a short sentence. Both are optional. *field1* can be used to identify the alert/module/problem source, and *field2* as a brief description of it. The data field shows the data received by the module.

By default, the log field is stored at `/var/log/pandora/pandora_alert.log`.

Note that the parameter number and sorting, and the location of the file can be changed anytime.

## 6.2 USER GRAPHICS

---

The user graphics, also called custom graphs, allow the user to define graphics of a variable size, which content values of different modules and different agents. This way you can compare graphically information from several sources. In Pandora FMS 1.3.1 you can see data overlapped, but modifying a line at `fgraph.php` you can see them stacked instead. There is no limit to the number of elements to view, but from five, the amount of information showed makes it difficult to understand it, unless you use big size images (800x600, etc).

Finally, to make easy the showed value comparison task, whenever you add an element to the graphic, you can multiply its value, to see them in a similar scale, to do so, use the "Factor" field.

REPORTING > GRAPH BUILDER

Source agent	endor	<a href="#">Get Info</a>	
Modules	cpu_sys		
Factor	1	Width	550
Render now	Yes	Height	210
Period	Day	View Events	No
			<a href="#">Add/Redraw</a>

Figure 34: Editing custom graphs

To create them you must follow the next steps:

1. Click on "Administration > Manage reports > Graph builder".
2. At the next screen, select the agent from which you want to represent the data, and click on "Get info", to obtain the moudles it has.
3. Once the module list is shown, choose one module and select other value, as the graph size, if there is any fator to normalize data the period used, and whether to show data events or not.
4. Click on "Add/Redraw".
5. Then the graph appears and we can add more modules to it util we finish the creation of the customized graph.
6. The bottom part of the screen, "Custom graph store" allows you to name he graph and store it to see it afterwards, without defining all the fields again.

Once the custom graphs are createt, they can be seen at "Operation > Reporting > Custom graphs".

Finally, we show you two examples of custom graphs.

Pandora FMS Graph - Last week

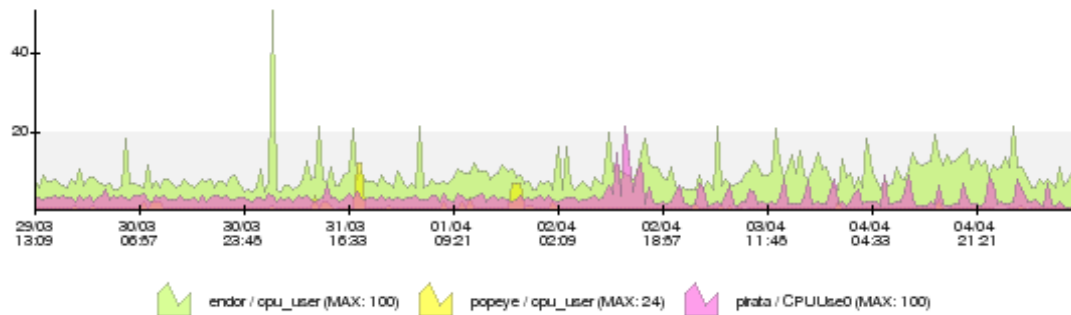


Figure 35: Last week activity graph

Pandora FMS Graph - Last 30 days

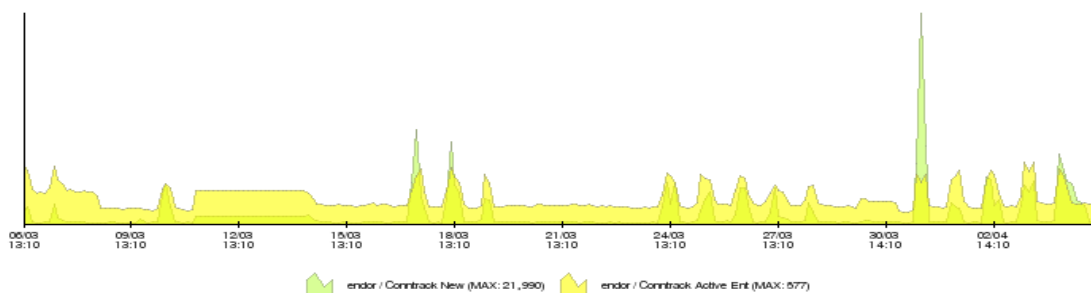


Figure 36: Last 30 days activity graph

## 6.3 CREATING CUSTOM REPORTS

Pandora FMS 1.3.1 allows you to create custom reports with that information you want from any agent. You can select alike user graphs different modules from different agents. The data is shown differently, depending on the kind of report we want to add:

- Simple graph: A simple graph, alike the ones shown at the agent data view. You can specify the period for the data.
- Custom graph: You can include any of the user graphs already defined.

- S.L.A.: You can specify a Service Level Agreement over the data in the specified period. You must also specify the minimum and maximum service values (i.e. if the valid latency time is 80ms, you can define a minimum of 0 and a maximum of 80). The valid % of the SLA, is that one from which the level of service is not accomplished, and it is red colored.
- Event report: Shows all the events of that agent in the specified period.
- Alert report: Shows all the alerts triggered by that module in the specified period.
- Monitor report: Works as a SLA over a valid service level of the monitor, and tells the percentage of time spent in each status.
- Avg. Value: Shows the arithmetic average value of the specified period.
- Max. Value: Shows the maximum value of the specified period.
- Min. Value: Shows the minimum value of the specified period.

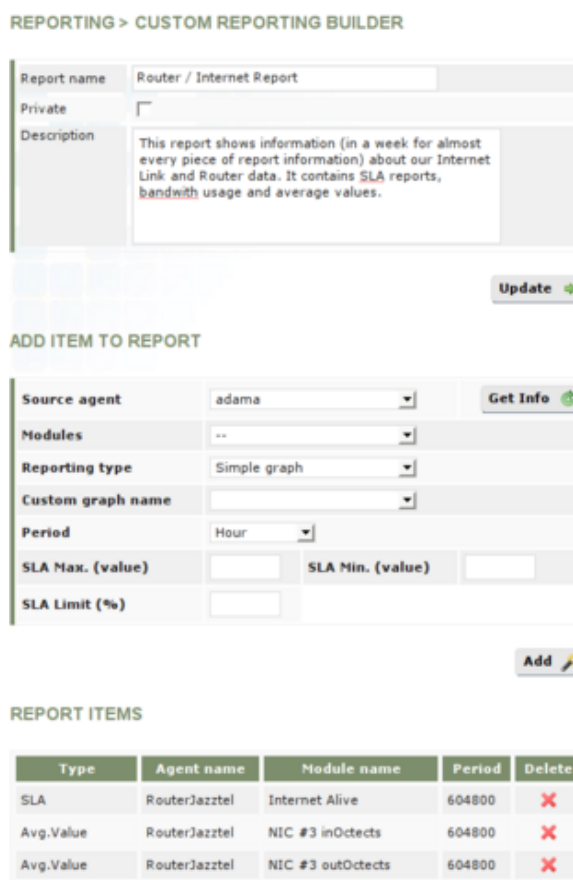


Figure 37: Report example

To create them you must procede as follows:

1. Click on "Administration > Manage Report > Report builder".
2. Click on "Add".
3. Enter the report name, whether is private or not, and a description.
4. Then the "Add item to report" box appears at the bottom of the report.
5. Add some types of elements from the ones mentioned before.
6. Clicl "Add" to add every part you need, each one with its own elements.

Once the reports are created, you can see them at "Operation > Reporting > Custom Reporting".

Viewing report example:

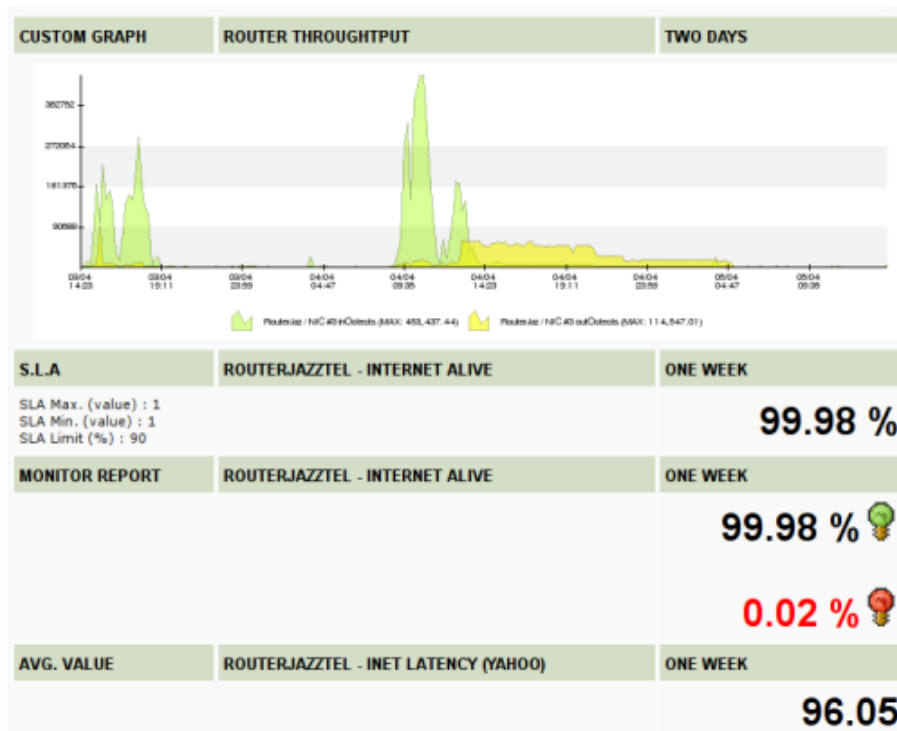


Figure 38: Report example

## 6.4 SETTING UP A VISUAL CONSOLE

Pandora FMS visual console is quite rude at the beginning, but using it with wisdom and a bit of patience, you can build complex views of your systems, and build your own views for each business/system process of your monitored data.

Also, if you want to create a "fullscreen" view, it is easy to modify code in `visual_console/render_view.php` file to render it in fullscreen, this will be probably a new feature for future releases.



Figure 39: Global view with a inherited item (Demo EU)

1. Create a new map in Administration -> Manage Reports -> Map Builder.
2. Create a map using a background that currently exists in Pandora FMS console. This maps are .png graphs that are placed on `/images/console/background/`. If you want to define your own backgrounds, simply place them there.

- Set the dimensions of the image, if are unknown, set 100x100, visualize the map results at "Operation -> Visual Console -> Your Map" and view the background properties. Use this information (width and height) to edit your map and set the correct dimensions.

Now you should add some data to this map. There are four types of items:

- Static images:** This represent boolean (on/off, good/bad) values of a specific monitor of a specific agent. This are called "static images" and are associated to a static image that are placed on `/images/console/icons`. You can place your icons there, using `xxx.png` for a "normal" icon, `xxx_bad.png` for a bad status, and `xxx_ok.png` for a good status.
- Line:** used to draw a line. It only uses "X", "Y", "Width" (as X2) and "Height" (as Y2) to draw a single line. If there are links among different items defined (as can be seen above), these lines are drawn automatically. If you want to create independent lines, this is the way to do it.
- Module Graph:** renders in a graph, the contents of a module of an agent, in the given interval. Use "Width" and "Height", "X" and "Y" to place the graph in the map where you want.

You can link any item in a static image to be clicked and take you to another map. This allows to setup interactive maps and also a easy browsing among views.

You can also establish "parental" relationships between objects in the same map, to represent lines linking items in a good or bad status. This lines are shown in red color if the parent is down, and in green color if is ok. This is very important.

Finally you can implment a monitoring hierarchy, based on other maps. You can hance a specific object in a map representing the "general" status of another previously defined map. This allows you to add or represent summarized information from hundred of modules using only one screen.

For example, using the global map you can set two icons in North America. These icons show a "global status" of the map they are linked to. If the map they represent has 10 elements, and all of them are green ("ok"), then this element will be shown in green as

ADD MAP ITEM			
Source agent	192.168.50.1	<a href="#">Get Info</a>	
Modules	--		
Reporting type	Static graph		
Period	Hour		
Position X	405	Position Y	100
Height		Width	
Label	Demo_EU		
Image	computer		
Map linked	Demo EU		
Parent item	firewall		
Label color	White	Link color	Yes

Figure 40: WorldMap - Creating link to Demo UE

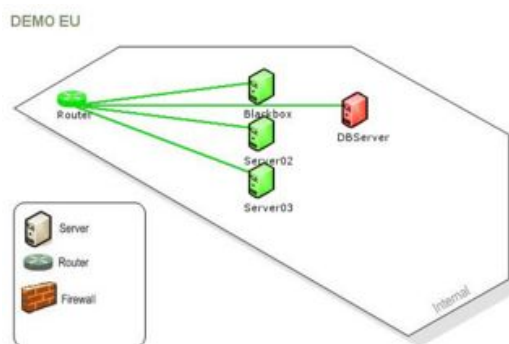


Figure 41: Demo EU Map

well, since everything is correct. But if any of those elements is red colored, then the icon will be shown in red, indicating there is something wrong inside. The user may click the object and see the full map being represented. This function can be recursive and have any link or detail depth wanted.

These elements represent other maps defined as "Static image" at the selection box, but they have no module attached to them, you must set the field "Map linked" to link this object to another map and show the "global" information about its status.

Try to add some data. After adding items, you can edit their positions, labels and images, if you want to change any field, you must delete and create it again.

## 6.5 MASSIVE COPY/DELETION/PROPAGATION OF MODULES AND ALERTS

Pandora FMS has a tool to massive copy/delete modules and alerts from an agent to another agent or list of agents. It is a very useful option when the user finds that modules and alerts configured for an agent would be repeated in a new agent. In order to simplify the administrator's work, Pandora FMS offers the option of copying modules and alerts defined in an agent, to be assigned to another. This is only valid for those modules as the Network modules which configuration is done in a centralized way. To copy software agents in Pandora FMS, is enough creating a identical file named *pandora\_agent.conf* and copy it to other systems.

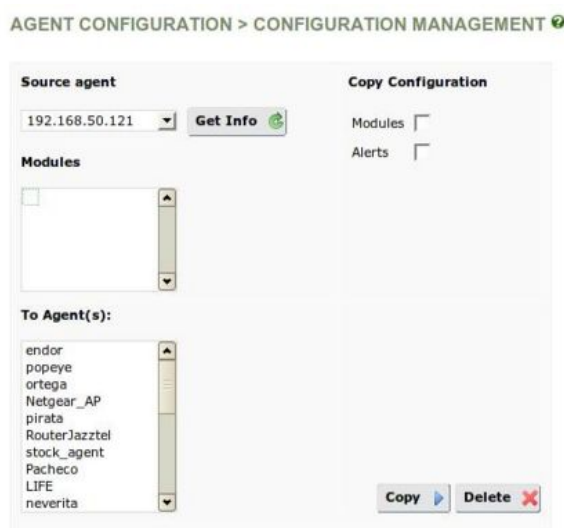


Figure 42: Configuration management

The screen is accessed through "Manage Agents > Manage Config.", in the Administration menu:

Source Agent menu allows the selection of the agent where the needed modules and/or alerts are.

The "Get Info" button shows the modules for that agent in the Modules list box.

- Copy process is performed to copy the module and/or alert configuration from the selected source agents to the selected destination agents. Several agents can be selected, pressing CTRL and the mouse right button simultaneously. The two check boxes at the top of the form are used to specify if the configuration to copy is from modules and/or from alerts. Every network agent copied will use the IP address of the objective agent to replace the source module IP address.
- Deletion process is performed to delete the configuration of the destination agents, in the multiple selection list box. Several agents can be selected at the same time, and the check boxes at the top of the form indicate whether it is the modules or the alerts configuration what is to be deleted. The application will prompt to confirm the deletion. Once the deletion is performed, the data associated to them will also be deleted.



## 7 MAINTENANCE AND TOOLS

---

### 7.1 PANDORA FMS MAINTENANCE

---

Pandora FMS infrastructure does not need maintenance, but it is very important to purge old data, and keep the database compacted. For this purpose there is a script called `/util/pandora_db.pl` in your Pandora FMS server package. This script performs all the database maintenance tasks:

- Deletes old data.
- Compacts existing data, interpolating them in several intervals, so the graphs will be the same, but the storage disk needed will be much less (one of the reasons why Pandora FMS is able to process that much information).
- Checks the database consistency for non existing modules, or for modules never used because they could not be initialized.
- Removes daily contact information with agents. Pandora FMS does not need more than 24hr of contact data history per agent, and if it is not purged, it slows down the Database.

This task should be done every night, and it is very important to do it, so please, take your time to understand this and setup the cron task. It is very easy and important to have Pandora FMS working fine.

To install the maintenance script on standard GNU/Linux systems, the following method is recommended:

1. Create a file named `/etc/init.d/pandora_db` containing the following lines:

```
#!/bin/bash
/usr/share/pandora_server/util/pandora_db.pl /etc/pandora/pandora_s
erver.conf
```

2. Change the file permissions:

```
chmod 750 /etc/init.d/pandora_db
```

3. Change the owner of the file:

```
chmod root:root /etc/init.d/pandora_db
```

4. Create a link to the cron dialy execution directory:

```
ln -s /etc/init.d/pandora_db /etc/cron.daily/pandora_db
```

From now on, every night the *script* will be executed, ensuring optimum status for the Database.

### 7.1.1 Installing cron job for Pandora FMS DB maintenance tool

As *root*, create a file at */etc/cron.daily* called "pandora\_db" with following content: As *root*, create a file at */etc/cron.daily* called "pandora\_db" with following content:

```
# Pandora FMS database maintenance
perl /usr/share/pandora/util/pandora_db.pl /etc/pandora/pandora_server.conf
```

Of course, you should have "*pandora\_db.pl*" and "*pandora\_server.conf*" at proper directories. If you have them at different places, you must change the command. Set permissions to 700 for this file:

```
chmod 700 /etc/cron.daily/pandora_db
```

And test it manually:

```
/etc/cron.daily/pandora_db
```

It should show you a message like:

```
Pandora FMS DB Tool 1.3 PS070828 Copyright (c) 2004-2007 Sancho
Lerena
This program is Free Software, licensed under the terms of GPL
License v2
You can download latest versions and documentation at
http://pandora.sf.net
Pandora DB now initialized and running (PURGE=60 days, COMPACT=15
days, STEP=1) ...

[PURGE] Deleting old data...
[COMPACT] Packing data from 2007-08-14 13:00:00 to 2007-08-21
13:30:26
[CHECKDB] Checking database consistency (step1)...
[CHECKDB] Checking database consistency (step2)...
[CHECKDB] Deleting non-init data...
```

## 7.2 PANDORA FMS DB STRESS TOOL

---

This is a small tool to test your Database performance. You can also use it to generate periodical or random data (using trigonometric functions) and fill fake modules.

You must create an agent and assign it some modules so that you can automatically insert data with this tool. The names must follow the notation below:

- *random*: to generate random data.
- *curve*: to generate a coincidence curve using trigonometric functions. Useful to see different interval interpolation work, etc.
- *boolean*: to generate random boolean data.

You can use any name containing the words "*random*", "*curve*" and/or "*boolean*", por

ejemplo:

- random\_1
- curve\_other

"data\_server" is the only **module type** you can select.

### 7.2.1 Pandora FMS DB Stress tool fine tune

This tool is preconfigured to search, in every agent, the modules named "random", "curve", or "boolean", which use an interval between 300 secs and 30 days.

If you want to modify this behaviour, then you have to edit the script *pandora\_dbstress* and modify some variables present at the beginning of the file:

```
# Configure here target (AGENT_ID for Stress)
my $target_module = -1; # -1 for all modules of that agent
my $target_agent = -1;
my $target_interval = 300;
my $target_days = 30;
```

The first variable line is *target\_module*, you must set it for a fixed module, or to -1 to process every matching target. The second variable line is *target\_agent*, for a specific agent. The third line is *target\_interval*, defined in seconds, and which represents the default periodical interval of the module. The fourth line is *target\_days* and represents the number of days in the past since the actual *timestamp*.

## 7.3 DATABASE MAINTENANCE

The core of Pandora FMS system is its Database. All the data collected by the monitored machines is stored in this database, from the administrator's data, to the events, incidents and audit data generated by the system at any time.

It is obvious that the efficiency and reliability of this module is vital for the proper functioning of Pandora FMS. A regular database maintenance is needed. To do so, the Database administrators can use standard MySQL commands. Maintaining Pandora FMS Database in good condition is

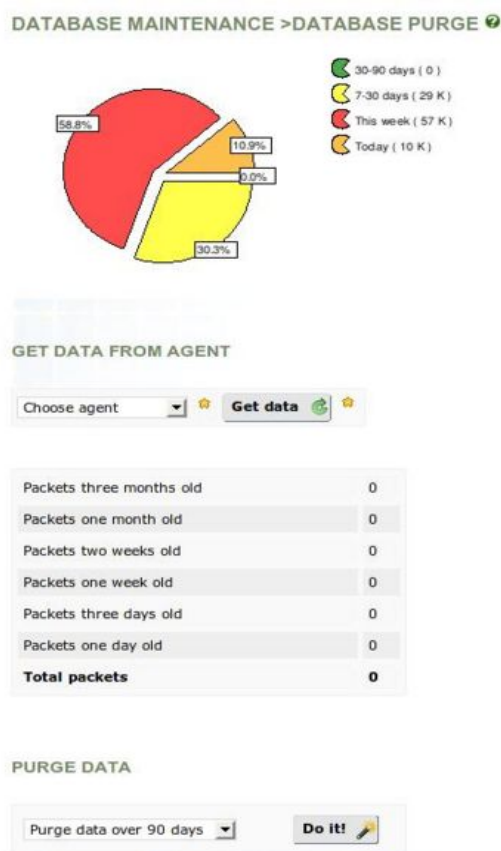


Figure 43: Purging database from console

critical for Pandora FMS to work properly.

As the database size increases linearly, the data will be compacted to reduce the amount of stored data without losing important information, especially the different graphs generated with the processed data. Going to "DB Maintenance", from the Administration menu, will show the Database configuration defined at the "Pandora Setup" option of the Administration menu to compact and delete data.

The DB statistics are generated by Agent, at "DB Maintenance > DB Information" in the "Administration" menu, and are represented in two kinds of graphs:

- Number of modules configured for each agent.
- Number of packages sent by each agent. A package is the group of data linked to the module that the agent sends each time interval.

All the out of range data received by an agent can be deleted from the "DB Maintenance > Database Purge" option at the "Administration" menu. The data is deleted through parameters from the "Delete data" screen shown in the graph above.

To delete data automatically, depending on Pandora FMS Settings option:

You must install Pandora DB script, as described at **Maintance section**

Max. days before compact data	<input type="text" value="15"/>
Max. days before purge	<input type="text" value="60"/>
Graphic resolution (1-low, 5-high)	<input type="text" value="5"/>
Compact interpolation in hours (1 Fine-20 bad)	<input type="text" value="1"/>

### 7.3.1 Manual purge of the Database

Pandora FMS has powerful tools for the administrator to manually purge the majority of the data stored in the Database. This includes data generated by both the agents, and the server.

#### **Agent's data purge. Debugging selected data from a module**

The option of purging the selected data from a module is used to eliminate those entries out of range, whatever the reason agent failure, out of range values, testing, DB errors, etc . Removing erroneous, incorrect or unnecessary data makes the graphical representation more accurate and shows the data without peaks or unreal scales.

From "DB Maintenance > Database Debug" at the "Administration" menu, any of the out of range data received from a agent's module can be deleted.

The purge settings are: "Source agent", "Modules", and "Maximum" and "Minimum" values out of range. Any parameter out of this minimum and maximum range will be deleted. For example, in a module registering the number of processes, if we are only

interested in values between 0 and 100, any values above that number will be usually produced by errors, noise or abnormal circumstances. If we set a range between 0 and 100, all those values below and above such as -1, 100 or 100000 will be permanently deleted from the database.

### 7.3.2 Normalizing data

It is possible to "clean" peaks and other disturbing information in modules by using the "normalize" icon (NEW in Pandora FMS 1.3), which removes data above or below (Data AVG)\* 1.5. You can find this option in the setup mode of every agent, selecting the "Modules" tab. Each module has an "Action" column, where you can find the icon.

This option is very useful when you have "peaks" in regular data, and this makes the graphic move to out of its scale limits. Use it carefully because it deletes data "outside" of the limits (above and below) the average value \* 1.5.



Figure 44: Normalizing data for a specific module

### 7.3.3 Database backup

A simple command, `mysqldump` will dump the contents of the Database. To restore the backed up data you will need an empty Database with the same name as the original one (usually "Pandora").

#### Making the backu

```
mysqldump -u root -p pandora > /backup/pandoradb_backup.sql
```

#### Restoring the original data

```
mysql -u root -p
create database pandora;
use pandora;
source /backup/pandoradb_backup.sql
```

You may also have to set permissions again for the new Pandora FMS console user:

```
grant all privileges on pandora.* to pandora@localhost identified by
```

```
'mypassword';
```

If you want to make a whole system backup, then you should not forget to backup the whole */etc/pandora* directory, to save the agent and server configuration information.

## 8 OTHER ADVANCED TOPICS

---

### 8.1 PANDORA FMS DATA SERVER VIRTUAL SERVERS

---

A special case to increase the servers processing power could be the usage of "virtual" servers. Using virtual servers (another example of the same server over the same machine) is recommended when Pandora FMS cannot process all the information without delay. Pandora FMS 1.3.1 uses a limited number of threads to process the information (this will change in future versions), therefore a solution against a very high load is to install another Pandora FMS Data Server (with another *incoming* directory), to be able to handle more information with the same machine.

### 8.2 PANDORA FMS DATABASE DESIGN

---

Pandora FMS first versions, from 0.83 up to 1.1, were based on a simple idea: a single data, a database insertion. This was very easy to develop and allowed easy searches, easy insertions and other operations.

This had many benefits but a big inconvenience: scalability. This system has a defined limit of modules that it can process, without implementing expensive clustering solutions that allow higher load, yet with a high amount of data (> 5 million items) the system was not that fast.

Solutions based on MySQL clustering are not easy and always add minor troubles, they also do not offer a long term solution.

Pandora FMS newest versions (1.3 and newer) implement a real time data compression for each insertion. It also allows to compress data using interpolation. Further more, it implements like in previous versions an automatic deletion of old data.

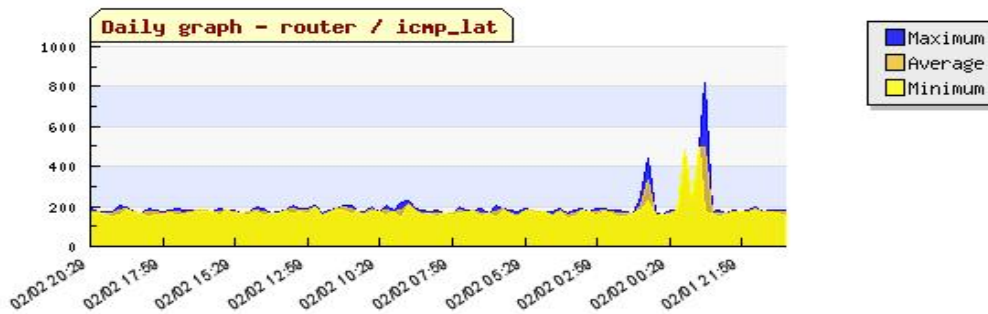
The new data processing system stores only "new" data. If a duplicated value is to be inserted, then the system rejects it and it will not be inserted. It is very useful to keep a reduced Database. This works for every Pandora FMS modules: numeric, incremental, boolean and string. In boolean data type the compactation index is very high, since are data that almost do not change. However "index" items are stored every 24 hours, so there is always a minimum amount of information that serves as a reference when compacting the information.

This solves part of the scalability problem, reducing the usage of the Database around 40%-70%. There is also another solution for scalability problems: the complete separation of every Pandora FMS components, allowing load balancing for file and data processing, and for executing network modules in different servers. Now you can have several Pandora FMS Servers (Network Servers, Data Servers or SNMP Servers) and Pandora FMS Web Console, as well as a Database or a high performance cluster (with MySQL5).

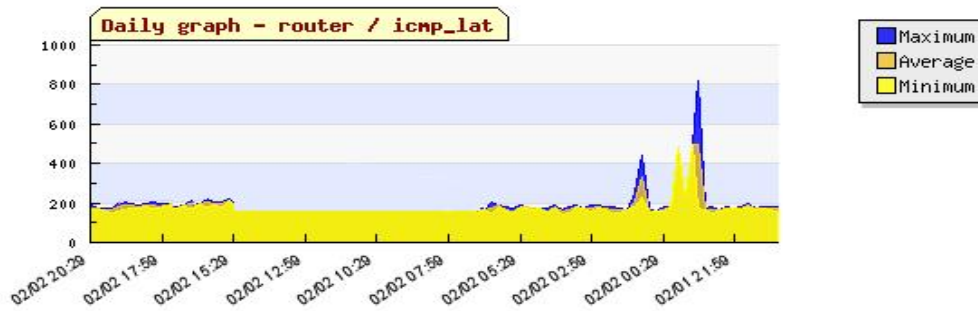
Modifications involve big changes to read and process data. The graphical engine has been redesigned and reimplemented completely, so that it can plot data very fast, according to the new data storage model. With this new version, if an agent cannot communicate with Pandora FMS, and Pandora FMS Server does not receive data from the agent, then this lack of data cannot have a graphical representation, so there will not be any change while plotting the graph.

You will be presented a perfect horizontal graph line. If Pandora FMS does not receive new data, then there is nothing to process, so everything will appear as before. It is similar MRTG behaviour.

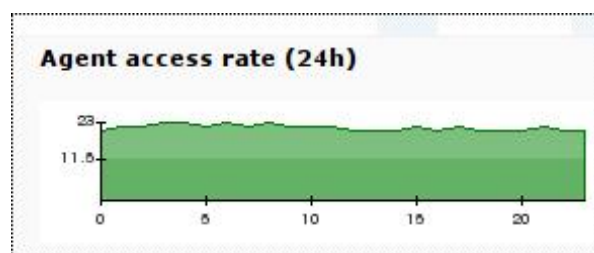
To see a graphical example, this image shows the change for each data, received every 180 seconds.



This would be the equivalent graph for the same data, except for a connection failure, from 05:55 a 15:29 aprox.



Pandora FMS 1.3 comes with a new general graph for the agent, which shows its connectivity and also the access ratio from its modules. This graph complements the other graphs that show when the agent has activity and is receiving data. This is an example of an agent regularly connecting to the server:

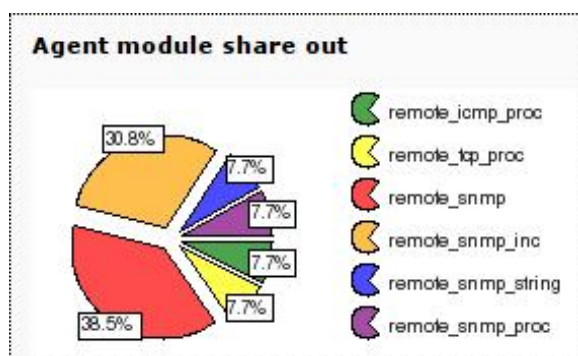




If it has (low) peaks in this graph, then it would mean that it has connection problems or high latency between the agent and Pandora FMS agent, or there is a connection problem from the Network Server.

### 8.2.1 Database index and other technical improvements

Small improvements to Pandora FMS Database relational model have been introduced. One of that changes is module type indexing. This way, accessing information is faster, since Pandora FMS logic agent, which gathers all the monitoring information, it is parted in different information pieces that come from very different sources. In next Pandora FMS version there will be up to four new specific servers so that it will be possible to process more information type.



Items like timestamp numeric representation (UNIX format), speed up date searches, date comparisons, etc. This work allowed a great improvement, reducing the time used in searches and insertions.

## 8.3 MySQL OPTIMIZATION FOR ENTERPRISE GRADE SYSTEMS

### 8.3.1 General advices

First of all, if you want to have a really HUGE system, that is, with tables bigger than 2GiB, MySQL recommends to use a 64Bit system. After this global recommendation, general rules are: the more RAM and CPU the better performance. According with our experience, RAM is more important than CPU. If you are planning to use 1GiB or a lower quantity of memory for your SQL system, please reconsider it. The minimum for enterprise-grade system, would be 2GiB. 4GiB is a good option for a big system. Keep in mind that more RAM can speed up key updates, keeping most of the used key pages in RAM.

Another good advice, if you are not using transaction-safe tables or you have large hdd and want to avoid long file checks, is to use a UPS. Then it is a good idea to be able to take the system down nicely in case of a power failure. For systems where the database is on a

dedicated server, you should look at 1G Ethernet. Latency is as important as performance.

Disk optimization is very important for very big databases: split databases and tables over different disks. In MySQL you can use symbolic links for this. Use different disks for system and database, and very important: try to use a low seek hard drive disk, because your application will be bound by the speed of your disk seeks, which increases by  $N \log N$ , as you get more data.

On GNU/Linux use `hdparm -m16 -d1` on the disks, at boot time, to enable reading/writing of multiple sectors at a time, and also DMA. This may increase the response time by 5-50%. Another good idea is to mount the disks with `async` (default) and `noatime`, this set of options does not update the file accessing time each time the files are used (read/write). For some specific application, you may want to have a RAM disk for very few specific tables, this could be a good option, but risky if there is a power failure and they are not saved into a non-volatile disk, so beware of that.

Use `--skip-locking` (default on some OS) if possible. This will turn off external locking and will give better performance.

If you run MySQL client and server on the same machine, use sockets instead of TCP/IP when connecting to MySQL (this can give you up to a 7.5% improvement). You can do this specifying no hostname, or localhost, when connecting to MySQL server. DISABLE binary loggin and replication if you are running only one MySQL server host.

### 8.3.2 Server setup variables

This configuration sample uses an example system with 4CPUs, 4GiB RAM, and it uses InnoDB tables for a double database setup, both with very heavy use. Keep in mind that total sum of memory reserved in each variable cannot be higher than 80% of total system memory. Try adjusting values for your setup, and keep an eye on MySQL logs at starting daemon.

```
# Example MySQL config file for very large systems (4GB, 4 CPU's)
[mysqld]
port                = 3306
socket              = /var/lib/mysql/mysql.sock
skip-locking
key_buffer          = 1000M
max_allowed_packet = 4M
table_cache         = 2048
sort_buffer_size    = 16M
read_buffer_size    = 32M
read_rnd_buffer_size = 32M
myisam_sort_buffer_size = 64M
thread_cache_size   = 64
query_cache_size    = 128M
# Try number of CPU's*2 for thread_concurrency
thread_concurrency = 8
# required unique id between 1 and 2^32 - 1
# defaults to 1 if master-host is not set
```

```
# but will not function as a master if omitted
server-id          =1
# Uncomment the following if you are using InnoDB tables
# You can set .._buffer_pool_size up to 50 - 80 %
# of RAM but beware of setting memory usage too high
innodb_buffer_pool_size = 1484M
innodb_additional_mem_pool_size = 60M
```

#### References:

- <http://dev.mysql.com/tech-resources/presentations/presentation-oscon2000-20000719/index.html>
- <http://jeremy.zawodny.com/mysql/mysql-optimization.html>

## 8.4 NTP UPDATE

---

It is very important to have synchronized all components (Servers, Database, Console). It is a good idea to have a cron script synchronizing every hour, something like:

```
ntpdate ntp.nasa.gov
```

## 8.5 SSH SERVER SECURIZATION

---

Pandora FMS uses sftp/ssh2 (scp) to copy the data files from the agents to the server. Because of that, it will need, at least, one data server with a SSH2 server listening to "pandora" user. This can be a quite important security risk, over a network which can be not secured. OpenSSH2 is **very** secure, but concerning informatic security there is nothing absolutely secure, therefore, you will have to take measures to make it "more" secure.

To securize SSH *scponly* is recommended, a small tool that bans remote logins using SSH. This way, you can ban remote SSH logins for "pandora", and only allow sftp/scp on that system.

### 8.5.1 What is *scponly*?

*Scponly* is an alternative 'shell' for system administrators that want to provide read and write permissions to remote users without providing any execution privilege. It could be described as a intermediate system between the system and the SSH applications.

The general use of *scponly* is creating a semi-public account, different than the concept of anonymous FTP account. This allows the system administrator to share files alike an FTP, but using the protection of SSH. This is especially important if you consider that FTP authentications cross public networks using a plain text format.

Using *scponly* to securize "pandora" user is very easy:

Installing *scponly* (for debian based systems):

```
apt-get install scponly
```

Or using `yum install scponly` with suitable repositories, or installing it manually with `rpm -i scponly`.

Replacing the "pandora" user *shell* for *scponly*:

```
usermod -s /usr/bin/scponly pandora
```

Done! With this, "pandora" user will be able to copy files with *scp*, but will not access the server as "pandora" user.

There is more information at [scponly web site](#).

## 8.6 FTP SERVER HARDENING (PROFTPD)

---

From version 1.3, it also supports all the platforms of its agent, you can use FTP to transfer the XML data files, therefore, you need a data server with a FTP server ready for "pandora" user. This could be a quite important security issue, in a network that needs to be securized.



This little recommendations to securize FTP, are for [proftpd](#) daemon, a highly configurable FTP server GPL licensed, that includes several options for access restriction.

It is recommended to set this parameters at **proftpd.conf**

```
Umask                077  077
MaxInstances          30
DefaultRoot /var/spool/pandora/data_in pandora
```

*DefaultRoot* uses *pandora* group, so you must create the group "pandora" and add the user "pandora" to it.

Another file that controls the access at level user is */etc/ftpusers*, in this file there are the names of those users not allowed to connect to this server.

```
[root@myserver]# cat /etc/ftpusers
```

```
root
bin
daemon
adm
lp
sync
shutdown
halt
```

```
mail
news
uucp
operator
games
guest
anonymous
nobody
```

Try to log in to FTP using "pandora" user and read directories other than */var/spool/pandora/data\_in* (this should be the only directory visible to this user).

## 8.7 VSFTPD HARDENING

---

*Vsftpd* has different parameters to securize a FTP account, but this can create conflicts with *scponly*. Some changes to improve the security of "pandora" account are recommended, so you can use FTP and SSH transfer systems at the same time.

1. Change the *home* directory of "pandora" to */var/spool/pandora/data\_in*
2. Keep *scponly* as the default shell.
3. Copy or move the directory */home/pandora/.ssh* to */var/spool/pandora/data\_in*. Do not forget to check that */.ssh* "pandora" user is the owner and it has de right permissions.
4. Modify *vsftpd* configuration file: */etc/vsftpd.conf* and add the following parameters:

```
check_shell=NO
dirlist_enable=NO
download_enable=NO
deny_file=authorized_keys
deny_file=.ssh
chroot_local_user=YES
```

This configuration sets "pandora" *home* directory to */var/spool/pandora/data\_in*, and does not allow "pandora" to connect remotely to stablish a interactive command session. It also allows FTP transfers with the same user, "pandora", to send files but it can only access the data incoming directory and does not allow to read nor list the content of any other directory or file.

## 8.8 INTEGRATING ALERTS WITH JABBER IM

---

It is very easy to set up Pandora FMS to send alerts through a Jabber server. Jabber can be a sistem to get real time alerts as well as a historic log, allowing a group of people to receive those alerts simultaneously.

### 8.8.1 Installing Jabber services

From the client side:

1. Install a Jabber client, like for example *Gaim* (now *Pidgin*).
2. Register an account (using *Pidgin*: configure the account clicking on "Accounts" tab).
3. Login that account.

From Pandora FMS Server side:

1. Install *sendxmpp*. With this tool you Pandora FMS can send messages to Jabber services.
2. Create the file *.sendxmpprc* inside the folder */home*.
3. Edit that file and insert the following text:

```
useraccount@jabber.org password
```

4. Change that file permissions:

```
chmod 0600 .sendxmpprc
```

Now you can send private messages using the command line, for example:

```
$ echo "Hello" | sendxmpp -s pandora useraccount@jabber.org
```

To register the alert at Pandora FMS Web Console, add a new alert and configure its variables. It is a good idea to do as follows:

- Field\_1: Jabber address.
- Field\_2: Text.

The alert will be defined as follows:

```
echo _field2_ | sendxmpp -s pandora _field1_
```

## 8.8.2 More examples of Jabber usage

Send a message to a chat room:

```
$ echo "Dinner Time" | sendxmpp -r TheCook --chatroom
test2@conference.jabber.org
```

Send the log lines to a Jabber destination, as they appear:

```
$ tail -f /var/log/syslog | sendxmpp -i
sysadmin@myjabberserver.com
```

NOTA: Be careful not to flood public Jabber servers or you can be banned from them.

## 8.9 USING IMAGE\_GRAPH (PEAR) WITH PANDORA FMS

---

The following text describes the installation and integration of *Image\_Graph* project from the source code, and its integration with Pandora FMS. However, there is also a modified version of *Image\_Graph* that comes with Pandora FMS Web Console, although some of

the problems that can appear are discussed in this section.

### 8.9.1 Introduction

Pandora FMS has a graphical engine which uses *ImageGraph* libraries to plot graphics. Pandora FMS has also its own graphical engine which uses *GDlib* directly.

### 8.9.2 Installation

If you want to install the last *ImageGraph* follow the next steps. It is not necessary to install *ImageGraph* externally, since Pandora FMS Web Console has a modified version of these libraries, licensed under GPL.

#### Installation on Debian GNU/Linux **unstable**

```
apt-get install php-pear php-image-canvas php-image-graph
```

Another form of installing it can be using directly the *pear* framework:

```
apt-get install php-pear

pear install --alldeps -f Image_Graph-alpha
pear install --alldeps -f Image_Canvas
pear install --alldeps -f Image_Color
```

### 8.9.3 Problems with Pear *Image\_Graph*

#### Undefined function: imageantialias()

Some systems report the error: *"PHP Fatal error: Call to undefined function: imageantialias() in /usr/share/php/Image/Canvas/GD.php"* It is a question related with some buggy PHP installations. There is some related info in [link](#) and there is also a [patch](#).

#### Undefined function: setclipping

You must apply the following patch (obtained from [\[1\]](#))

```
diff -u pear/Image_Canvas/Canvas.php:1.5 pear/Image_Canvas/Canvas.php:1.6
--- pear/Image_Canvas/Canvas.php:1.5      Fri Sep 30 14:59:35 2005
+++ pear/Image_Canvas/Canvas.php         Sun Nov 27 17:22:28 2005
@@ -25,7 +25,7 @@
 * @author      Jesper Veggerby <pear.nosey@[...].dk>
 * @copyright   Copyright (C) 2003, 2004 Jesper Veggerby Hansen
 * @license     http://www.gnu.org/copyleft/lesser.html  LGPL License 2.1
- * @version    CVS: $Id: Canvas.php,v 1.5 2005/09/30 18:59:35 nosej Exp
$
+ * @version    CVS: $Id: Canvas.php,v 1.6 2005/11/27 22:22:28 nosej Exp
$
 * @link        http://pear.php.net/pepr/pepr-proposal-show.php?id=212
 */

@@ -587,6 +587,20 @@
```

```

function image($params)
{
}
+
+ /**
+  * Set clipping to occur
+  *
+  * Parameter array:
+  *
+  * 'x0': int X point of Upper-left corner
+  * 'y0': int Y point of Upper-left corner
+  * 'x1': int X point of lower-right corner
+  * 'y1': int Y point of lower-right corner
+  */
+ function setClipping($params = false)
+ {
+ }

```

### 8.9.4 Problems with PHP *Safemode*

When PHP *Safemode* is activated, then the fonts used and other referenced files from Pandora FMS cannot be symbolic links. Pandora FMS includes a *truetype* font to use it with the Web Console. If you modify this font or use another one, you can have problems for this reason. Permissions can also be a problem while working with the safe mode (**Safemode**). All PHP components must have the same permissions and owners to avoid problems.

Note for Debian/Ubuntu users: To install more *truetype* fonts at the directory */usr/share/fonts/truetype/freefont*, you can use the following command:

```
apt-get install ttf-freefont
```

### 8.9.5 Interesting links

- [Image Graph official web page](#)
- [Many examples](#)
- [Image Graph forum](#)

## 8.10 SMS GATEWAY

---

This section describes how to set a SMS sending gateway based on sending queue. This way you can implement a SMS sending server, connected to a cell phone and sending the SMS through Gnokii software, and different remote servers, can send your SMS messages so that the server can process them. This allows different Pandora FMS Servers (or other machines that want to use the gateway) to use send messages in a centralized way, without using one cell phone for each server.

First, you must create the user "sms" in the host which will act as the SMS gateway. Then create the directories *home/sms* and */home/sms/incoming*. If you want to use the SMS



gateway from other machines, you need to make the directory */home/sms/incoming* accessible to other servers using any file transfer or sharing method: NFS, SMB, SSH (scp), FTP or Tentacle.

The mechanism of the SMS sending gateway is easy: Each file inside the directory */home/sms/incoming* will be processed, then deleted, and a SMS with the contents of the file will be sent. This file must have a specific format, detailed as follows:

***Phonenumber/SMSText***

### 8.10.1 SMS Gateway implementation

You must create four scripts:

**SMS:** Script that sends SMS using Gnokii and a cell phone connected with a USB data cable. This script must be only in the system where the gateway is (the system that has the data cable connected to the GSM cell phone).

**SMS\_GATEWAY:** Script that periodically processes the incoming directory (*/home/sms/incoming*), processing pending files. This script must be only in the system where the gateway is.

**SMS\_GATEWAY\_LAUNCHER:** Launcher script for SMS\_GATEWAY script (start stop daemon). This script must be only in the system where the gateway is.

**COPY\_SMS:** It copies an SMS using the command **scp** from the client computer to the gateway system. It uses the PHONE\_NMBR as the first parameter, and the text as the second one, using "" to specify each parameter. The script trust on automatic SSH authentication and on user "sms" for the transfers. You can replace "scp" for "cp" at the local system, or use another system like Tentacle to transfer the file.

#### **sms**

This is the script that sends SMS using Gnokii. Gnokii must be correctly configured (using the file */etc/gnokii.conf* or similar). You may have to be *root* to launch the script, otherwise set *SETUIDO* bit on gnokii binary.

```
#!/bin/bash
texto=$1
number=$2
if [ $# != 2 ]; then
echo "I need more parameters"
exit 1;
fi
/bin/echo $1 | /usr/local/bin/gnokii --sendsms $2
```

#### **sms\_gateway**

This is the daemon script for the *gateway*:

```
#!/bin/bash
```

```

INCOMING_DIR=/home/sms/incoming
HOME_DIR=/home/sms

while [ 1 ]
do

    for a in `ls $INCOMING_DIR`
    do
        if [ ! -z "$a" ]
        then
            NUMBER=`cat $INCOMING_DIR/$a | cut -d "|" -f 1`
            MESSAGE=`cat $INCOMING_DIR/$a | cut -d "|" -f 2`
            TIMESTAMP=`date +%Y/%m/%d %H:%M:%S`
            echo "$TIMESTAMP Sending to $NUMBER the message
$MESSAGE" >> $HOME_DIR/sms_gateway.log
            $HOME_DIR/sms "$MESSAGE" "$NUMBER"
            echo "$TIMESTAMP Deleting $a" >>
$HOME_DIR/sms_gateway.log
            rm -Rf $INCOMING_DIR/$a
            sleep 1
        fi
    done
    sleep 5
done

```

### **sms\_gateway\_launcher**

This is the launcher script for *sms\_gateway*:

```

#!/bin/bash

# SMS Gateway, startup script
# Sancho Lerena, <slerena@gmail.com>
# Linux Version (generic)

# Configurable path and filenames
SMS_GATEWAY_HOME=/home/sms
SMS_PID_DIR=/var/run
SMS_PID=/var/run/sms.pid

# Main script

if [ ! -d "$SMS_PID_DIR" ]
then
    echo "SMS Gateway cannot write it's PID file in $SMS_PID_DIR.
Please create directory or assign appropriate perms"
    exit
fi

if [ ! -f $SMS_GATEWAY_HOME/sms_gateway ]
then
    echo "SMS Gateway not found, please check setup and read manual"

```

```

        exit
    fi

    case "$1" in
        start)
            OLD_PATH="`pwd`"
            if [ -f $SMS_PID ]
            then
                CHECK_PID=`cat $SMS_PID`
                CHECK_PID_RESULT=`ps aux | grep -v grep | grep
"$CHECK_PID" | grep "sms_gateway" | wc -l`
                if [ $CHECK_PID_RESULT == 1 ]
                then
                    echo "SMS Gateway is currently running on this
machine with PID ($CHECK_PID). Aborting now..."
                    exit
                fi
            fi

            nohup $SMS_GATEWAY_HOME/sms_gateway > /dev/null 2> /dev/null & 2>
/dev/null > /dev/null
            sleep 1

            MYPID=`ps aux | grep "$SMS_GATEWAY_HOME/sms_gateway" | grep -v
grep | tail -1 | awk '{ print $2 }'`
            if [ ! -z "$MYPID" ]
            then
                echo $MYPID > $SMS_PID
                echo "SMS Gateway is now running with PID $MYPID"
            else
                echo "Cannot start SMS Gateway. Aborted."
            fi
            cd "$OLD_PATH"
            ;;
        stop)
            if [ -f $SMS_PID ]
            then
                echo "Stopping SMS Gateway"
                PID_2=`cat $SMS_PID`
                if [ ! -z "`ps -F -p $PID_2 | grep -v grep | grep
'sms_gateway'`" ]
                then
                    kill `cat $SMS_PID` 2> /dev/null > /dev/null
                else
                    echo "SMS Gateway is not executing with PID $PID_2, skip
Killing step"
                fi
                rm -f $SMS_PID
            else
                echo "SMS Gateway is not running, cannot stop it."
            fi
            ;;
        force-reload|restart)
            $0 stop
    esac

```

```
    $0 start
    ;;
*)
    echo "Usage: sms_gateway {start|stop|restart}"
    exit 1
esac
```

### copy\_sms

This small script creates a file to send SMS in a client machine and copies it to the SMS gateway using *scp*:

```
#!/bin/bash

SERIAL=`date +%j%M%s`\
SERIAL=`hostname`_${SERIAL}

TEL=$1
TEXT=$2

echo $TEL\|$TEXT >> /tmp/${SERIAL}
scp /tmp/${SERIAL} sms@192.168.1.1:/home/sms/incoming
rm -rf /tmp/${SERIAL}
```

## 8.10.2 HP Network Node Manager (NNM) integration with Pandora FMS (SNMP)

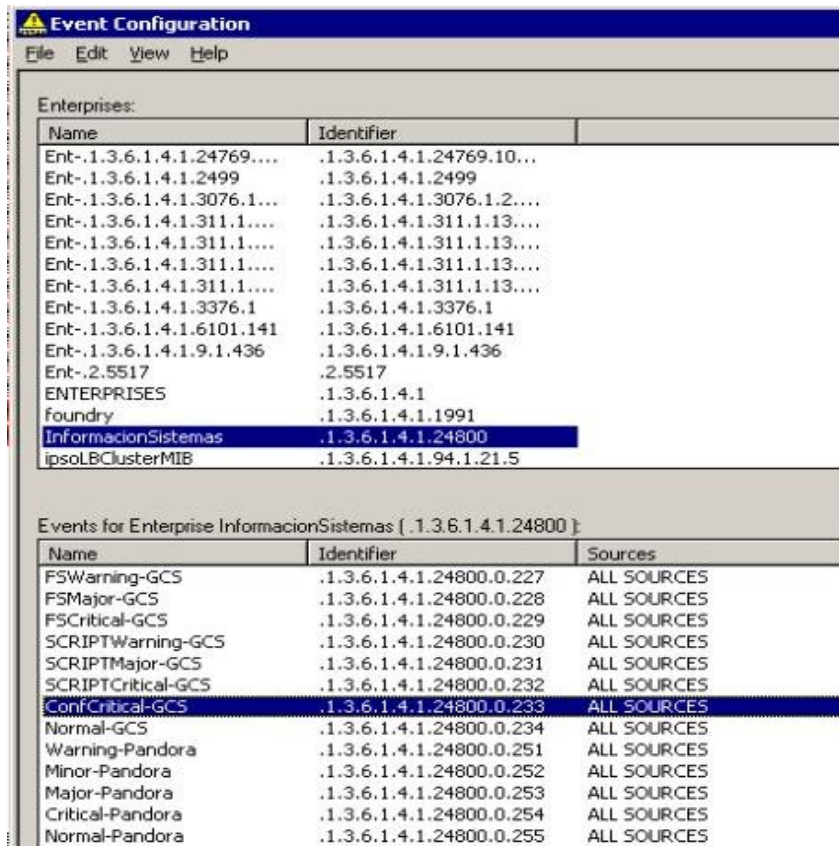
It is possible to use Pandora FMS along with HP Network Node Manager (NNM) to report all those important events gathered by Pandora FMS to a global system managed by HP OpenView Network Node Manager. This is useful since it allows Pandora FMS to monitor certain environments and send only those important events to another monitoring system, such as NNM.

Pandora FMS way of triggering alerts is very flexible when exporting certain data through SNMP traps. This will be the chosen mechanism to do the integration. You will have to create a script that will allow, parsing some arguments, to generate specific traps that Network Node Manager will be able to interpret.

### Configuring events with NNM



First, we have to establish some "classifications" based on Pandora FMS events criticality. There will be five criticality levels (Normal, Warning, Minor, Major y Critical). We will define five types of events associated with a specific user OID, which in this case will be from 1.3.6.1.4.1.24800.251 to 1.3.6.1.4.1.24800.255). This are the attached OID to events of those five categories:

### Creating specific Pandora FMS alerts



From a single alert, with different parameters we will be able to export the alerts to NNM. To do so, we will create an alert as follows:

## ALERT CONFIGURATION &gt; MODIFY ALERT

Alert name	SNMP Trap
Command	/usr/share/pandora/snmp_trap_nnm.sh _field1_ _field 
Description	<div style="border: 1px solid #ccc; padding: 5px;"> <p>_field1_ SPECIFIC_OID _field2_ ERROR _field3_ DESCRIPCION</p> <pre>snmptrap -v1 -c public 192.168.127.66 .1.3.6.1.4.1.00248_agent_6_field1+_field2_ ".1.3.6.1.4.1.00248.0 &lt; field3</pre> </div>
<b>Update</b> 	

**SNMP trap generation script**

This alert has an attached script detailed below, and each server that processes and executes the exported module must call it:

```
#!/bin/bash

HOST=`hostname`
ESPECIFIC_OID=$1          # _field1_ pandora
ERROR=$2                 # _field2_ pandora
DESCRIPCION=$3          # _field3_ pandora
HOST_DESTINO=nnm
COMMUNITY=public
ENTERPRISE_OID=.1.3.6.1.4.1.24800
WARNING=0
MINOR=1
MAJOR=2
CRITICAL=3
NORMAL=4
MAXL=2000
DIR=`dirname $0`
LOGFILE="$DIR/`basename $0` | awk -F'.' '{print $1}'`.log"
CONF="$DIR/`basename $0` | awk -F'.' '{print $1}'`.conf"

truncate_log()
{
  typeset LOG=$1
  typeset LINES=$2
  typeset TMP1=/tmp/$PRG.tmp1

  if [ -z "$LOG" ] ; then
    echo "$PRG: truncate_log(): Parametros incorrectos -> [$LOG] [$LINES]"
    return
  fi

  if [ ${LINES%[kK]} != $LINES ] ; then # Log binario, truncamos con dd

    LOGSIZ=$(du -k $LOG | awk '{print $1}')
    MAXSIZ=${LINES%[kK]}                # Eliminamos caracter 'k' final
    typeset -i SKIP                      # Nm. de bloques de 1K a obviar
                                         # Ver comando 'dd' abajo
  fi
}
```

```

if $LOGSIZ -gt $MAXSIZ ; then
echo -n "$LOG : Truncando a $MAXSIZ Kbytes ..."
(( SKIP = LOGSIZ - MAXSIZ ))
dd if=$LOG of=$TMP1 bs=1040 skip=$SKIP
if $? -eq 0 ; then
#
# Vaciado de $LOG
#
>$LOG
if $? != 0 ; then
echo "$PRG : Error vaciando fichero $LOG"
return
else
cat $TMP1 >>$LOG
if $? != 0 ; then
echo "Error rellenando $LOG"
return
fi
echo 'OK!'
>$TMP1
fi
else
echo "Error en comando 'dd if=$LOG of=$TMP1 bs=1024 skip=$SKIP'"
fi
fi
else
# Log ASCII, truncamos con tail
NUMLIN=$(wc -l <$LOG)
if $NUMLIN -gt $LINES ; then
echo -n "$LOG : Truncando a $LINES lineas ..."
(( NUMLIN -= LINES - 1 ))
tail +$NUMLIN $LOG >$TMP1
if $? -eq 0 ; then
#
# Vaciado de $LOG
#
>$LOG
if $? != 0 ; then
echo "$PRG : Error vaciando fichero $LOG"
return
else
cat $TMP1 >>$LOG
if $? != 0 ; then
echo "Error rellenando $LOG"
return
fi
echo 'OK!'
>$TMP1
fi
else
echo "Error en comando 'tail +$NUMLIN $LOG >$TMP1'"
fi
fi
fi
rm -f $TMP1
} # truncate_log()
#
# MAIN
#
# Crea log si no existe
#
[ ! -f $LOGFILE ] && touch $LOGFILE

for linea in `cat $CONF | grep -v "^#"`

```

```

do
  if [ ! -z `echo $linea | grep "HOST" ` ]
  then
    HOST_DESTINO=`echo $linea | grep "HOST" | awk -F=' ' '{print $2}'`
  elif [ ! -z `echo $linea | grep "COMMUNITY" ` ]
  then
    COMMUNITY=`echo $linea | grep "COMMUNITY" | awk -F=' ' '{print $2}'`
  elif [ ! -z `echo $linea | grep "ENTERPRISE_OID" ` ]
  then
    ENTERPRISE_OID=`echo $linea | grep "ENTERPRISE_OID" | awk -F=' ' '{print
$2}'`
  elif [ ! -z `echo $linea | grep "WARNING" ` ]
  then
    WARNING=`echo $linea | grep "WARNING" | awk -F=' ' '{print $2}'`
  elif [ ! -z `echo $linea | grep "MINOR" ` ]
  then
    MINOR=`echo $linea | grep "MINOR" | awk -F=' ' '{print $2}'`
  elif [ ! -z `echo $linea | grep "MAJOR" ` ]
  then
    MAJOR=`echo $linea | grep "MAJOR" | awk -F=' ' '{print $2}'`
  elif [ ! -z `echo $linea | grep "CRITICAL" ` ]
  then
    CRITICAL=`echo $linea | grep "CRITICAL" | awk -F=' ' '{print $2}'`
  elif [ ! -z `echo $linea | grep "NORMAL" ` ]
  then
    NORMAL=`echo $linea | grep "NORMAL" | awk -F=' ' '{print $2}'`
  elif [ ! -z `echo $linea | grep "LOG" ` ]
  then
    LOG=`echo $linea | grep "LOG" | awk -F=' ' '{print $2}'`
  else
    LINEA=`echo $linea`
  fi
done

if [ "$ERROR" = "WARNING" ]
then
  ESPECIFIC_OID=`expr $ESPECIFIC_OID + $WARNING`
elif [ "$ERROR" = "MINOR" ]
then
  ESPECIFIC_OID=`expr $ESPECIFIC_OID + $MINOR`
elif [ "$ERROR" = "MAJOR" ]
then
  ESPECIFIC_OID=`expr $ESPECIFIC_OID + $MAJOR`
elif [ "$ERROR" = "CRITICAL" ]
then
  ESPECIFIC_OID=`expr $ESPECIFIC_OID + $CRITICAL`
elif [ "$ERROR" = "NORMAL" ]
then
  ESPECIFIC_OID=`expr $ESPECIFIC_OID + $NORMAL`
else
  if [ $LOG ]
  then
    echo ""
    echo " ---- Error. Nivel de error $ERROR no especificado ---- "
    exit 1
  fi
fi

#echo "$WARNING $MINOR $MAJOR $CRITICAL $NORMAL"
#echo "$COMMUNITY $HOST_DESTINO $ENTERPRISE_OID $HOST $ESPECIFIC_OID
$DESCRIPCION"

if [ $LOG ]
then
  echo ""

```



```
echo ""
echo "COMMUNITY=$COMMUNITY ENTERPRISE_OID=$ENTERPRISE_OID ESPECIFIC_OID=
$ESPECIFIC_OID DESCRIPCION=$DESCRIPCION" >> $LOGFILE
echo "/usr/bin/snmptrap -v1 -c $COMMUNITY $HOST_DESTINO $ENTERPRISE_OID $HOST
6 $ESPECIFIC_OID $ENTERPRISE_OID.0 s '$DESCRIPCION'" >> $LOGFILE
fi

/usr/bin/snmptrap -v1 -c $COMMUNITY $HOST_DESTINO $ENTERPRISE_OID $HOST 6
$ESPECIFIC_OID $ENTERPRISE_OID.0 s "$DESCRIPCION"

if [ $? -eq 0 ]
then
echo "Ejecucion correcta" >> $LOGFILE
echo ""
else
echo "No se ha ejecutado correctamente" >> $LOGFILE
echo ""
fi

#
# Trunca el fichero de log si supera MAXL lineas
#
truncate_log $LOGFILE $MAXL
```

### 8.10.3 Using the alert on a module

This is a screenshot of how to use the alert, defining an alert on a module.

The field *field1* is used to store the OID and to know to which group of alarms is attached (in our example we will use "251", defined at NNM). The field *field2* is the criticality degree (when sending the trap, MAJOR represents a number added to 251 to form the OID). The field *field3* is the description of the alarm.

**ALERT ASSOCIATION FORM**

Alert type	SNMP Trap	Alert status	Enabled
Min. Value	1.00	Max. Value	1.00
Alert text			
Description	Serial 0/0/0:0 primaria caida		
Field #1 (Alias, name)	251		
Field #2 (Single Line)	MAJOR		
Field #3 (Full Text)	_timestamp_ Agente: _agent_ Serial 0/0/0:0 primaria caida		
Time from	00:00	Time to	00:00
Time threshold	30 minutes	Other	
Min. number of alerts	6	Max. number of alerts	1
Assigned module	Serial 0/0/0:0 1801		

### 8.10.4 Visualizing data in NNM

And this is the final view of Network Node Manager console, receiving events from Pandora FMS:

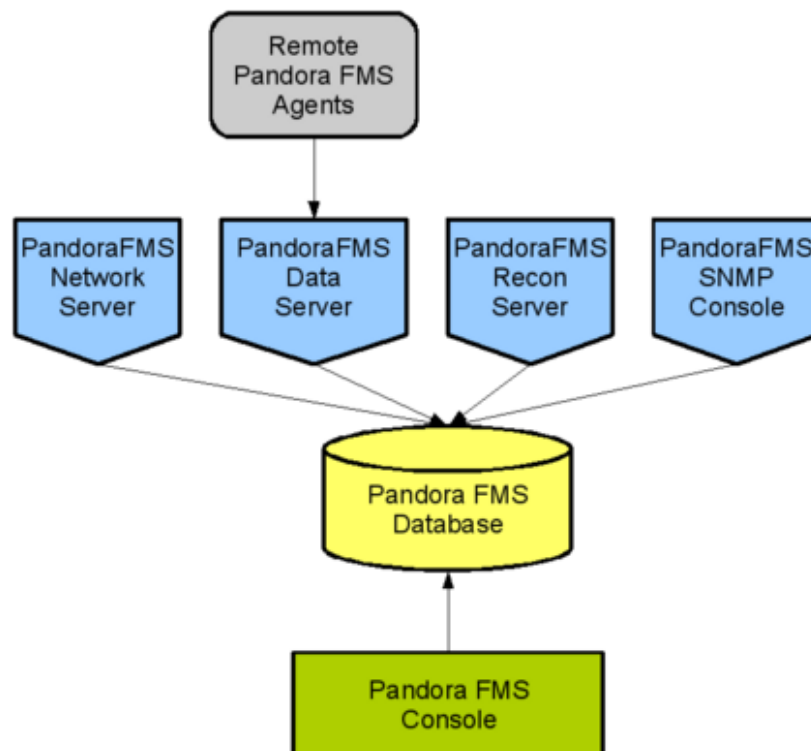
Ack	Corr	Severity	Date/Time	OID-Name	Source	Message
		Block	Fri 02 23 19:39:00	pandora	2008-03-23 19:39:00	Agente: 60 serie Serial 0/1/1 secundaria 60000 caida
		Block	Fri 02 23 19:40:00	pandora	2008-03-23 19:40:00	Agente:latencia 60000 Latencia: 47.92
		Block	Fri 02 23 19:41:00	pandora	2008-03-23 19:41:00	Agente:latencia 60000 Latencia: 46.56
		Warning	Fri 02 23 19:44:07	pandora	2008-03-23 19:44:07	Agente : 600000000 Sesiones : 3100.00
		Block	Fri 02 23 19:49:27	pandora	2008-03-23 19:49:26	Agente: 600000000 Serial 0/1/0 secundaria 60000 caida
		Block	Fri 02 23 19:50:43	pandora	2008-03-23 19:50:41	Agente: 600000000 Uso excesivo de CPU valor: 100.00
		Warning	Fri 02 23 19:50:47	pandora	2008-03-23 19:50:46	Agente : 600000000 Sesiones : 584.00
		Warning	Fri 02 23 19:52:15	pandora	2008-03-23 19:52:15	Agente : 600000000 Sesiones : 3983.00
		Block	Fri 02 23 19:52:15	pandora	2008-03-23 19:52:16	Agente:latencia 60000 Latencia: 36.14
		Warning	Fri 02 23 19:52:27	pandora	2008-03-23 19:52:27	Numero conexiones 600000000 2995.00 definidas(100/2500)
		Block	Fri 02 23 18:01:31	pandora	2008-03-23 18:01:31	Agente:latencia 60000 Latencia: 02.96
		Block	Fri 02 23 18:10:00	pandora	2008-03-23 18:10:01	Agente: 600000000 Serial 0/1/1 secundaria 60000 caida
		Block	Fri 02 23 18:11:44	pandora	2008-03-23 18:11:43	Agente:latencia 60000 Latencia: 53.34
		Block	Fri 02 23 18:11:44	pandora	2008-03-23 18:11:43	Agente:latencia 60000 Latencia: 39.41
		Warning	Fri 02 23 18:14:01	pandora	2008-03-23 18:14:01	Agente : 600000000 Sesiones : 606.00
		Warning	Fri 02 23 18:14:01	pandora	2008-03-23 18:14:01	Agente : 600000000 Sesiones : 3167.00
		Block	Fri 02 23 18:14:01	pandora	2008-03-23 18:14:01	Agente:latencia 60000 Latencia: 30.32
		Block	Fri 02 23 18:20:00	pandora	2008-03-23 18:20:00	Agente:latencia 60000 Latencia: 45.10
		Block	Fri 02 23 18:20:13	pandora	2008-03-23 18:20:11	Agente:latencia 60000 Latencia: 14.62
		Block	Fri 02 23 18:21:14	pandora	2008-03-23 18:21:14	Agente:latencia 60000 Latencia: 37.80
		Warning	Fri 02 23 18:21:00	pandora	Declarar conexiones alta 1000-03-23 18:21:07 exceed: 7.00	
		Block	Fri 02 23 18:24:14	pandora	2008-03-23 18:24:14	Agente : 600000000 Sesiones : 221.00
		Block	Fri 02 23 18:24:40	pandora	2008-03-23 18:24:40	Agente:latencia 60000 Latencia: 72.77
		Warning	Fri 02 23 18:29:00	pandora	2008-03-23 18:29:00	Numero conexiones 600000000 2979.00 definidas(100/2500)
		Block	Fri 02 23 18:30:12	pandora	Declarar conexiones alta 1000-03-23 18:30:12 exceed: 7.00	
		Block	Fri 02 23 18:30:01	pandora	2008-03-23 18:30:01	Agente: 60 serie Serial 0/1/0 secundaria 60000 caida
		Block	Fri 02 23 18:30:07	pandora	2008-03-23 18:30:07	Agente : 600000000 Sesiones : 510.00
		Warning	Fri 02 23 18:44:43	pandora	2008-03-23 18:44:43	Agente : 600000000 Sesiones : 3150.00
		Warning	Fri 02 23 18:45:00	pandora	2008-03-23 18:45:00	Numero conexiones 600000000 3054.00 definidas(100/2500)
		Block	Fri 02 23 18:46:07	pandora	2008-03-23 18:46:06	Agente:latencia 60000 Latencia: 34.71
		Block	Fri 02 23 18:46:14	pandora	Numero procesos exceed alto 2008-03-23 18:46:14 exceed: 0.00	

## 9 HIGH AVAILABILITY

### 9.1 PANDORA FMS'S HIGH AVAILABILITY FEATURES

Due to all the tests, features and bugs reported by the users, PandoraFMS is a very stable application nowadays, though, when facing critical or high load environments, it is necessary to ensure that if one of any PandoraFMS's components fails the whole system will not suffer any consequence that is to say, that PandoraFMS will keep running as if nothing happened. PandoraFMS has been designed to be a modular application and any of its modules can work by its own. Aside from that, it also has been designed to be able to work along with other components and it is able to takeover the task of a failed component.

PandoraFMS's standard design could be the one showed in the following picture.



*Figure 45: Standard implementation of PandoraFMS without HA*

Clearly, agents are not redundant. If an agent fails, it does not make much sense to run another one instead, most likely it failed because it is being impossible to gather information coming from a module cause its execution is being done correctly therefore, this could not be fixed launching another agent yet again running in parallel besides, it could be too because the system has no communication or it has crashed. The solution is pretty obvious, to redund critical systems beside whether they have PandoraFMS's agents thus to

replicate the monitoring of those systems.

Aside from the console, which redundancy is pretty easy to apply, it just requires two recon servers with alternatives tasks. This way, if one falls, the other one will take over of its task and will keep it running. It is possible to enable *HA* in several scenarios:

### 9.1.1 MySQL Cluster

It is possible to configure a database cluster to set at the same time either *HA* and load balancing. **Database is the most critical component** of the whole architecture , therefore the cluster is the best solution. It is only a matter of converting the DB schema in MySQL cluster compatible tables. This picture has been proven and it works fine, but an advance knowledge regarding to MySQL5 cluster would be desirable, plus, the fact of give the nodes a good amount of RAM. Minimum 2GiB in a two nodes architecture with a maximum of 5000 modules (in total)

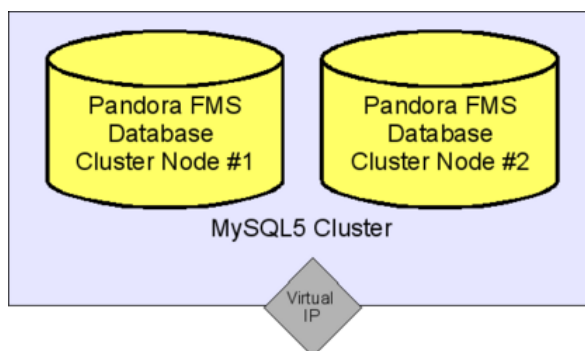


Figure 46: Two MySQL cluster nodes

In this case, it is not necessary a special setup of PandoraFMS.

### 9.1.2 Multiple PandoraFMS's consoles

In this case, no special configuration is needed either. It is easy, it is only needed to install another console. Either of them can be used to access the information, no matter where they are. By using a web balancer would be possible to reach any of them in a transparent way, not knowing which one is being used at an specific moment because the session management system uses *cookies*, which is kept in the browser. To do this we have used *LVS* to load balancing and *KeepAlived* to do the *HA* all of this process has been described later on.

### 9.1.3 HA over PandoraFMS data server

This is the most complex scenario, regarding to PandoraFMS, no special configuration is required, although, another tool is needed to implement the *HA* and the load balancing, there are quite a few commercial tools that do *HA* and balancing, there are also *OpenSource* solutions such as *vrpdp*, *LVS* or *Keepalive*.

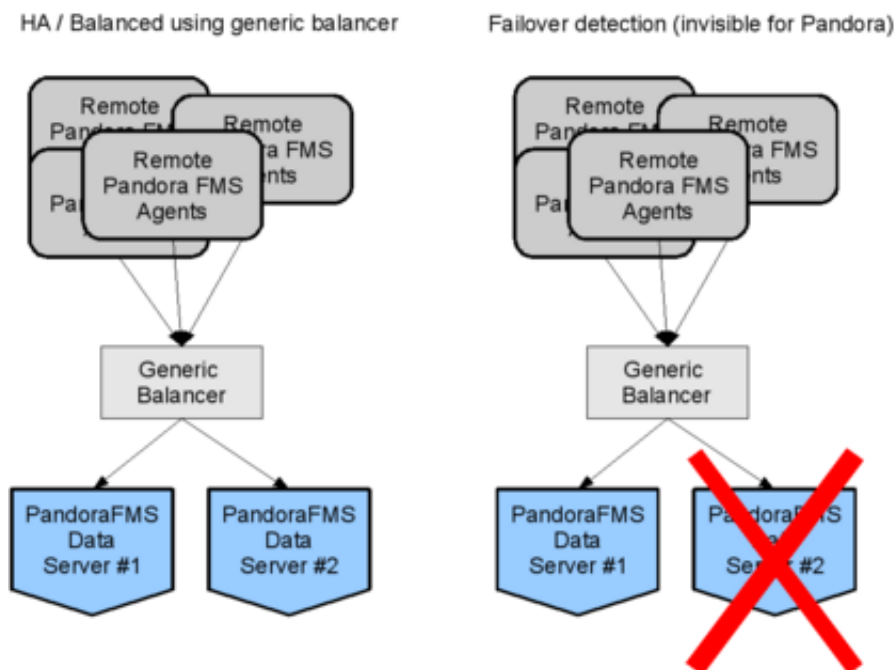


Figure 47: HA implemented over two nodes running PandoraFMS data server and a balancer.

For the PandoraFMS data server, two identical machines would be needed, with the same public keys for all the agents that would connect to the machine, also, it is needed to duplicate the master key of the SSH server (if ssh is used as a default way to communicate the agents with the server). Using tentacle the process is much easier, all you would need would be to replicate the configuration. Each machine would have a different IP, and the balancer would (as happens with the MySQL cluster) one unique IP where the agents would send all their data. The task of sending the data to whatever server its correspond is done by the balancer.

If one of them fails, due to the *HA*, the agents would keep sending the data to the same IP they did before the fall, without noticing the fact that one of the server is actually not running. No special configuration is required in the PandoraFMS data server, moreover, each server can keep its own name, this can be useful to identify (in the Server Status view) which one has fallen. PandoraFMS data modules can be processed by any server, no preassignment is required. It has been designed like this intentionally in order to make the *HA* implementation easier.

At the end of the chapter how to implement *HA* is described along with load balacing using *LVS* and *Keepalive* over a TCP service that could be either Tentacle port (41121), SSH one, FTP one or any other one. The same procedure can be use to build a cluster with 2 or more systems over an Apache Web Server to give access to PandoraFMS Web Console.

### 9.1.4 HA over PandoraFMS network server

This is simpler. It is required to install multiples network server in several machines of the network (all of them have to be able to reach the systems that are going to be monitor), besides, all of them have to be in the same segment (otherwise, latency data will not be coherent).

Network server can be marked as primaries. This server, will take over of the modules assigned to a server that has fallen, automatically. PandoraFMS servers include it own mechanism to detect if one of them has fallen, this is done by checking the last time it contacted (*server threshold x 2*). It is enough for one active PandoraFMS server to detect if all of them has fallen. If all the server fall, there is no way to detect or implement HA.

It is obvious that in a two nodes system, the best way to implement HA and load balancing is to assign the 50% of the modules to each server and mark both of them as masters (*Master*). If there were more than two master servers and a third server, which fell with pending modules, the first of the master server would takeover the modules and execute it. If the fallen server is recovered, the modules which actually belongs to it will come back automatically to it.

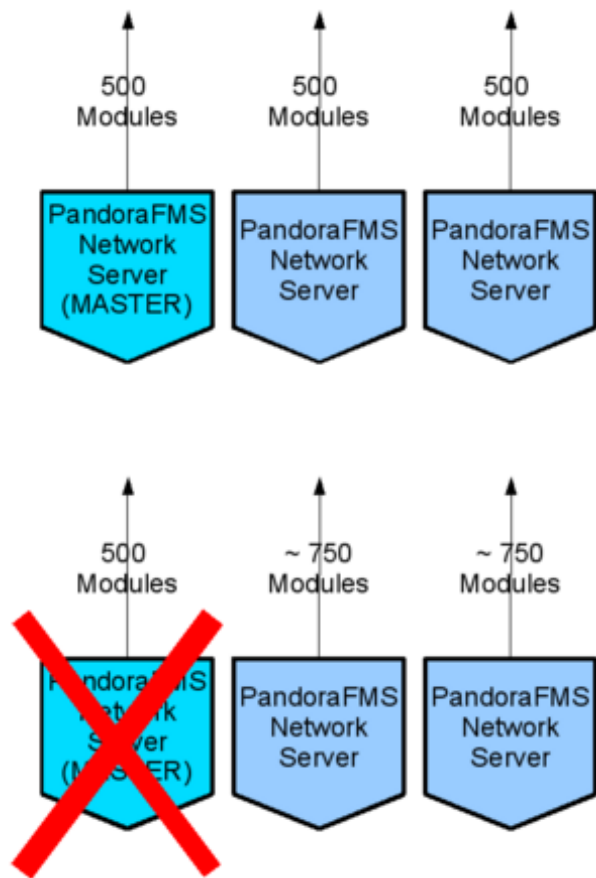


Figure 48: HA implementada con dos nodos del servidor de red de Pandora FMS

## 9.2 HA AND LOAD BALANCING WITH LVS AND KEEPALIVED

---

For the load balancing we are going to use **Linux Virtual Server** (LVS). To manage High Availability (HA) between services (SSH and Tentacle), we are going to use Keepalived.

**LVS:** is an advanced load balancing solution for Linux systems, its mission is to build a high-performance and highly available server for Linux using clustering technology, which provides good scalability, reliability and serviceability.

**IPVS:** implements transport-layer load balancing inside the Linux kernel, so called Layer-4 LAN switching. IPVS is incorporated into Linux Virtual Server, where it runs on a host acts as a load balancer at the front of a cluster of real servers, it can direct requests for TCP/UDP based services to the real servers, and makes services of the real servers to appear as a virtual service on a single IP address

**Keepalived:** It is used to manage LVS. *Keepalived* is being used in the cluster to keep under control SSH or Tentacle servers, either in the PandoraFMS data server node 1 or the PandoraFMS data server node 2, to check whether SSH and Tentacle Server are alive or not. If one of those services fall, Keepalived will let LVS knows that one of the two nodes has fallen and it has to redirect the connections to the other node which is alive.

*Keepalived* has been chosen as a HA service because it has "sessions persistence", which is going to be used by the servers. For instance, if one of the nodes falls, the users that might be working in that said node, will be redirected to the other node which is alive, they will not notice any change, neither in their current sessions nor in their sessions (SSH will not work due to its own implementation of the encrypted sessions, but any other TCP connections, such as SSL or FTP, will work just fine). Particularly in Tentacle/SSL, there will be an attempt to reconnect and the information in the data package will not be lost.

*Keepalived* configuration file as well as *Keepalived* commands are described in Appendix 2.

**Load Balancing algorithm:** There are two algorithms widely know nowadays: «Round Robin» and «Weight Round Robin», both are quite similar and are based in assigning jobs turn by turn.

Round-robin is one of the simplest scheduling algorithms for processes in an operating system, which assigns time slices to each process in equal portions and in order, handling all processes without priority. Round-robin scheduling is both simple and easy to implement, and starvation-free. Round-robin scheduling can also be applied to other scheduling problems, such as data packet scheduling in computer networks.

Weight-based load balancing improves on the round-robin algorithm by taking into account a pre-assigned weight for each server

This does not make any sense in the topology we are working on, since both machines has exactly the same hardware. Thus, we have chosen «Round Robin» as an algorithm.



### 9.2.1 What to do when a node fails

*Keepalived* will detect if any of the services goes down, removing it from the active LVS node list, so that all the requests sent to that module will be redirected to another active node.

Once the problem with the service down is solved, *keepalived* must be restarted:

```
/etc/init.d/keepalived restart
```

Restarting the service will make all the nodes to be reinserted in the available LVS node list.

If one of the nodes goes down, there will be no need to manually insert the nodes using *ipvsadm*, since *Keepalived* will do it, once restarted, and checked that the services that should be giving HA service are running and are accessible through their "HealthCheckers".

## 9.3 APPENDIX 1. LVS LOAD BALANCER CONFIGURATION

---

*ipvsadm* usage:

Installing *Linux Director* with *ipvsadm*:

```
ipvsadm -A -t ip_cluster:22 -s rr
```

Las opciones son:

The options are:

-A add a service

-t TCP service, format IP:port

-s Scheduler, se the parameter "rr" (round robin)

Install the nodes (real servers) to which the requests are to be redirected to port 22.

```
ipvsadm -a -t ip_cluster:22 -r 192.168.1.10:22 -m
```

```
ipvsadm -a -t ip_cluster:22 -r 192.168.1.11:22 -m
```

*ipvsadm* status without active connections is the following:

```
Prot LocalAddress:Port Scheduler Flags
-> RemoteAddress:Port      Forward Weight ActiveConn InActConn
TCP cluster:www rr
-> nodo-2:ssh              Masq    1        0        0
-> nodo-1:ssh              Masq    1        0        0
```

Using "Round Robin" algorithm both machines will have the same weight inside the cluster. Therefore, they will share the connections. You can see here an example of LVS balancing connections against the cluster:

```
Prot LocalAddress:Port Scheduler Flags
```

```

-> RemoteAddress:Port      Forward Weight ActiveConn InActConn
TCP cluster:ssh rr
-> nodo-2:ssh              Masq     1         12         161
-> nodo-1:ssh              Masq     1         11         162

```

## 9.4 *KEEPALIVED* CONFIGURATION

---

*Keepalived* is in charge of checking the services at your configuration file (*/etc/keepalived/keepalived.conf*).

To start *Keepalived*:

```
/etc/init.d/keepalived start
```

To stop *Keepalived*:

```
/etc/init.d/keepalived stop
```

The configuration file used for the cluster is the following:

```

# Configuration File for keepalived
global_defs {
    notification_email {
        email@valido.com
    }
    notification_email_from keepalived@domain
    smtp_server 127.0.0.1
    smtp_connect_timeout 30
    lvs_id LVS_MAIN
}
virtual_server 192.168.1.1 22 {
    delay_loop 30
    lb_algo rr
    lb_kind NAT
    protocol TCP
    real_server 192.168.1.10 22 {
        weight 1
        TCP_CHECK {
            connect_port 22
            connect_timeout 3
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    real_server 192.168.1.11 22 {
        weight 1
        TCP_CHECK {
            connect_port 22
            connect_timeout 3
            nb_get_retry 3
            delay_before_retry 1
        }
    }
}

```

## 10 WHAT IS NEW IN PANDORA FMS 1.3

---

Pandora FMS 1.3 was freed the fourth quarter of 2007. Pandora FMS 1.3.1 was freed at the end of April, 2008. Basically 1.3.1 is a enhanced version of 1.3 but without big changes: there are many bugfixes and some new features detailed below.

### 10.1 PANDORA FMS 1.3

---

- New main view (tactical).
- New group view (faster, more usable for operators).
- New visual console. Allows to create custom hierarchical views with graphical components.
- Whole console graphical redesign. This includes tab navigation and other important changes for usability and better layout.
- New custom graph maker, allows to join in the same graph items from any agent in the same graph.
- New report builder. It manage reports with different componentes (SLA, Graph, Event report, max, min, avg. values, etc).
- Improved incident support (could be generated from Recon Server also)
- Site news, to get production sites up to date for working teams.
- New logo ! :-)
- All graphs are generated now from special version of Pear ImageGraph (redistributed with Pandora FMS). No more jgraph dependencies.
- Support for new Pandora FMS recon server, able to manage network sweep with unlimited sensors. New hosts are detected and monitored immediatly.
- Editor for new network components items. Network Components are predefined network modules that administrator could use to define more quickly the monitorization schema.
- Editor for new network templates. This is a kind of predefined "meta agent", composed by many network components. An agent could be assigned to a network template and inherit all defined modules as its own. Also new hosts found with Pandora Recon Server could be assigned to a network template.
- Hosts could have more than IP address now.
- Pandora FMS now supports different templates (using CSS).
- Big performance improvement in parallel work with console thanks to an improvement in session management.
- Big performance improvement in all graphic generation because a rewrite of core graphical functions.
- Many improvements in graphical system.
- Alerts now support text alerting using regular expressions.
- Alerts now supports time limits (from time to time).
- Fully integrated upgrade tool to upgrade from 1.2 version.
- Able to manual validate alerts from console. This returns to "green" status for alert, and generate an event trace in event viewer (validated by current user).

- More support for HP-UX (August 2007).
- Added option to renice pandora progress (only Unix/Linux agents).
- Added option to have a warm\*up time in startup phase (only Unix/Linux agents).
- Added option to setup a user defined codepage (only Unix/Linux agents).
- Added option to define file copy method (ftp, ssh, local) (only Unix/Linux agents).
- Added installer for textmode installation (only Unix/Linux agents).
- Better debug.
- Fixed important stability problems (Windows only) from 1.2 version.
- Support for custom server port in SSH (only Unix/Linux agents).
- Added FTP support to windows agent and unix agents.
- Better installer for Windows.
  
- New Recon Server to detect new systems and begin monitoring them.
- Upgraded Network Server, now works with more threads, this improves the performance **A LOT** compared with 1.2 version.
- New code thread-safe for all networking core.
- ICMP Proc checks makes several ping checks (defined by user) before consider target is down.
- Alerts expired/out of timegap/recovered are displayed in event viewer and set to green status automatically.
  
- Tested on MySQL clustering enviroments.
- Specific performance hints for Pandora FMS.
- **Great** performance improvement on graphs.
- Better database management, it purges non initialized modules and malformed modules.
  
- Tested on MySQL clustering enviroments.
- Specific performance hints for Pandora FMS.
- **Great** performance improvement on graphs.
- Better database management, it purges non initialized modules and malformed modules.
  
- Support for RPM packaging system (SuSe, Fedora, Redhat).
- Support for DEB packaging system (Debian, Ubuntu).

## 10.2 PANDORA FMS 1.3.1

---

### 10.2.1 New documentation

Is not really new, but about 60% of current documentation is new stuff and the other 40% has been revised. Added a lot of images and schemes and translation has been revised.

### 10.2.2 Tentacle

Pandora FMS 1.3.1 more important change is that **Tentacle** [\[1\]](#) is now the official transport method for Pandora FMS Agents. New agent versions are released, but SSH compatibility

is maintained, so you don't need to change any agent if don't want to. Tentacle is a new protocol that supports SSL and X509 standard, multiplatform and client-server architecture, is used by Pandora FMS to copy XML data files from agents to [Pandora FMS Data Server](#) in a secure way.

### 10.2.3 New server options

Pandora FMS 1.3.1 comes with several new server options:

#### Alert recovery notification

If you want to optionally activate alert ceased notification by sending the same alert but with "[RECOVERED]" text added in field2 and field3, you should apply a new token in server config file (pandora\_server.conf)

```
alert_recovery 1 | 0
```

Defines if Pandora FMS launch another alert when alert condition is recovered. It has the same field1, but adds "[RECOVER]" to field2 and field3. Is disabled by default.

#### TCP and SNMP specific timeout time

TCP specific options :

- tcp\_checks: number of tcp retries if first attempt fails. 2 by default
- tcp\_timeout: specific timeout for tcp connections. 30 by default

SNMP specific options :

- snmp\_checks: number of snmp request retries if first attempt fails. 1 by default
- snmp\_timeout: specific timeout for snmp request. 10 by default.
- snmp\_proc\_deadresponse: Return "down" status if cannot connect or that module return invalid response (like NULL) for remote\_snmp\_proc. This kind of requests are usually for interface status. By default is enabled.

All these parameters, especially the last ones, which allow to adjust the network checks, allow at a deployment with different Network Servers, at different network parts, to establish different timeouts and retries to fit better in different network environments: specific WAN Network Servers with high latency times and packet loss, LAN Network Servers with minimum latency times and without packet loss, servers that work with slow SNMP devices, etc.

### 10.2.4 Other new features

- Added agent search filter in agents view (operation and administration mode).
- Now user is able to validate (Cease) manually an alert (new feature).

- Added a automatic installer for Pandora FMS servers.
- .RPM and .DEB Packages for all components (including tentacle).
- Full documentation in Spanish and English (PDF). Online documentation is also present.
- Created a Nagios to Pandora FMS migration tool.

### 10.2.5 Bugs fixed

Pandora FMS 1.3.1 has fixed a lot of bugs detected in 1.3 version

- (Console) Fixed bug with combined graphs added in reports. More of two graphs make contents to repeat in each combined graph.
- (Console) Fixed bug with combined graph: scale are now overlaped not stacked.
- (Console) Fixed several bugs in tactical view and in server view.
- (Console) Improved usability in several pages adding tooltips.
- (Console) Some fixes in visual console for image sizes, added tooltips.
- (Console) Fixed several problems with ACL / Group access.
- (DB Core, Servers, Console) Make a reimplementacion of keepalive. Code rewritten.
- (Console) Fixed problem in combined graph. Much better render now.
- (DB Core) Generic\_data\_inc data with float numbers are now managed ok.
- (Console) https support for console.
- (Console) Fixed small bug mixing min/max values of SLA render.
- (Console) Fixed small bug deleting item from report, that return user to another report.
- (Console) Solved small visual error with filter. Added Logs text (Internal audit viewer).
- (Console) Security fix has been included in ver\_agente.php, using a new function to validate GET variables (checking for numeric data). This security issue was a SQL Blind URL Attack.
- (DB Core) Insert\_event is now capable to store events already validated.
- (Console) Added additional ACL check. Now user is able to validate (Cease) manually an alert (new feature).
- (Console) Fixed problem in visual maps, alerts was not used to draw red icons. Fixed.

- (Console) Solved bug in lagcheck for down modules (tactical, server view).
- (Core) Removed detection of GNU/Linux in startup (not a bug exactly...)
- (Network server) Fixed problems for snmp\_proc module. Tested on heavy load systems.
- (DB Core) Fixed bug for float data in snmp\_data modules and negative values. Code cleanup and implemented a new method to get data.
- (Server utils) Fixed compaction script. Bug in compaction function that delete data insted compacting it :-). (pretty effective eh?).
- (Network server) Better management of thread locking. This should fix latests problems reported.
- (Data server) Minimal improvement on zero data files.
- (SNMP Console) Some small fixes reported from unknown user.
- (Network server) Minimal optimization in locking for SNMP threads.
- (Network server) Fixed problem with default retries of SNMP library.
- (SNMP Console) Fixed parameter for snmptrapd call (%a instead %B). This was causing problems (do not detect IP Address) on Redhat systems.
- (DB Core) Fixed alert that fires always one time more than max\_alert defined.