Security Architecture

Security Architecture

Security Architecture

The purpose of this document is to describe the security elements of each Pandora FMS component, so that the administrator knows them and knows how to use them to implement a more secure architecture, in accordance with regulations such as PCI / DSS, ISO 27001, ENS, LOPD or similar. In addition, this file provides a specific description of the security mechanisms of each Pandora FMS element, possible risks as well as the way to minimize them, using the tools available in Pandora FMS or other possible mechanisms.

DECLARACIÓN DE CONFORMIDAD CON EL
_ens
Esquema Nacional de Seguridad
Categoría BÁSICA
RD 3/2010

Security implementation (general)

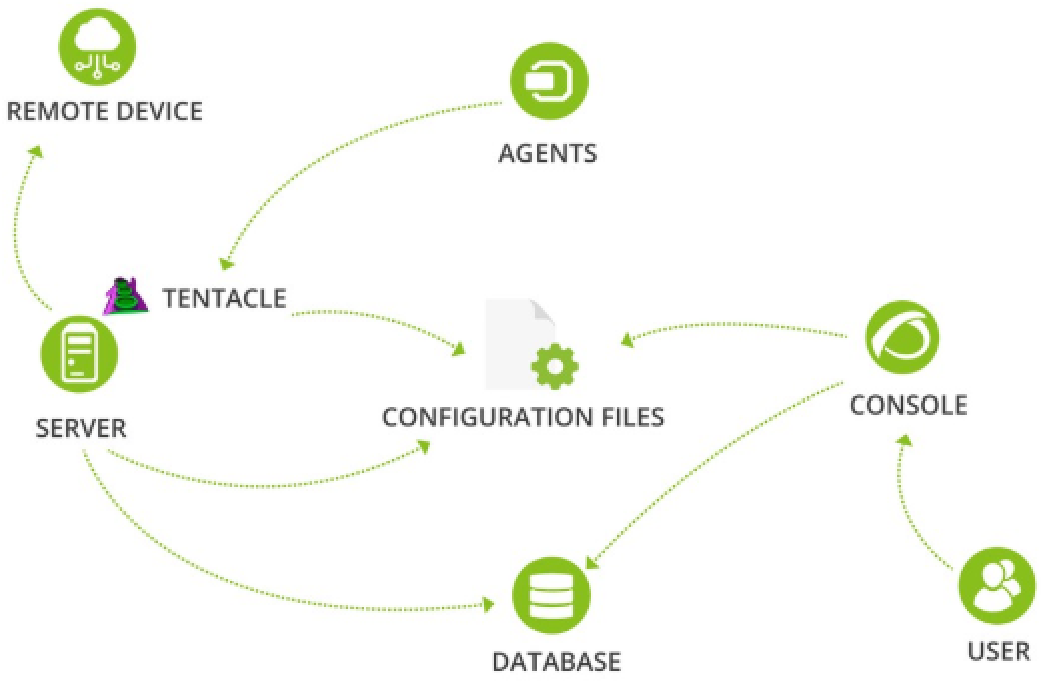
These points apply to international standards such as PCI / DSS, ISO 27001, National Security Scheme, LOPD, etc. They work as a guide for a safe Pandora FMS implementation in your environment.

- Pandora FMS components have their input and output ports documented, so it is possible to secure all accesses to and from their components by means of Firewalls.
- Safe traffic through encryption and certificates: Pandora FMS supports SSL / TLS encryption and certificates at both ends and at all levels (user operation, communication between components).
- Dual access authentication system: A double authentication system can be implemented. The first one is placed at access level (HTTP) integrated with any open source or commercial token system.
- Authentication with third parties system: located at the application level, it is managed by Pandora FMS, which can be authenticated against LDAP or Active Directory.
- SSO (Single Sign-On), with SAML.
- Security policies in user management: User management is defined by policies both at user profile level and at operational visibility profile level, defined as the Enterprise version Extended ACL system.
- Possibility of audits on the actions of the monitored elements: Pandora FMS Enterprise version audits all user actions, including information or altered or deleted fields. In addition, it includes a validation system by signing these records.

- Audit data transfer to external log managers: Audit logs are available for export through SQL and allow them to be integrated into a 3rd source for higher security, almost in real time.
- Physical separation of components that offer an interface to the user and the information containers (filesystem). Both the data stored in the database and the filesystems that store monitoring configuration information can be in separate physical machines, in different networks, protected through perimeter systems.
- Active password policy that enforces a strict password management policy for users to access the application (console).
- Sensitive data encryption. The system allows the most sensitive data such as login credentials to be stored in an encrypted and secure manner.

Security by architecture components

Pandora FMS architecture, in a very simple way, can be summarized as follows:



Server

- The server needs root permissions, but it can be installed (with certain limits) with non-root permissions (Linux systems only).
- The server needs direct access (read and write) to the agent remote configuration files of the agents, which are spread when the agents contact the server periodically. These files are protected by the filesystem, with standard permissions.
- The server itself does not listen to any port. It is the Tentacle server who listens on a port, the server only accesses the files left by the Tentacle server on disk.

- The server has its own very detailed log.
- The server connects to the main database using a standard MySQL / TCP connection.
- Part of the code is accessible (OpenSource) and that of the enterprise version can be requested under specific contract conditions (for customers only).

Possible vulnerabilities and safeguards

- Unauthorized access to agent configuration files. Solution:
    1. Implement an external secured container for external configuration files through NFS.
- Command injection on remote agents through the manipulation of configuration files stored in the configuration container. Solution:

1. Disable remote configuration on highly sensitive agents after configuration and leave them running without being able to carry out any changes remotely, for total security.
2. Remote monitoring - without agents - of the most delicate devices.

- Vulnerable against false information attacks, such as simulating agents that are not in the system or impersonating their identity. To avoid this, several mechanisms can be used:

1. Password protection system (which works by group).
2. Limiting agent self-creation, and creating them instead manually.
3. Limiting the ability to auto detect changes in the agent and not take new information from the XML, apart from the existing one.

- Malicious capture of communication between server and console (network traffic capture). Solution:
    1. Activate TLS communication between server and MySQl database.

Tentacle

- Tentacle is an official internet service, documented as such by IANA. This means that it can be easily protected with any perimeter security tool.
- It does not need root or special privileges.
- It has four security levels: No encryption (default), SSL / TLS Basic, SSL / TLS with certificate at both ends and SSL / TLS with certificate and CA validation.
- Specifically designed not to give clues to possible intruders in error messages and with specific timeouts to prevent brute force attacks.
- It has its own audit log.
- 100% of the code is accessible (under opensource GPL2 licence).

Possible vulnerabilities and safeguards

- Attacks on the filesystem. The configuration container must be accessed. Solutions:

1. It is protected in the same way as the server, by means of a secured external NFS system.

- DoS attacks due to overload. Solutions:

1. Set up an HA solution on the TCP service it offers for balancing, or an active / active cluster. Any

hardware or software solution available is valid because it is a standard TCP service.

Console

- It does not need root, it is installed with a user without privileges.
- It must have access to the agent configuration repository (filesystem).
- It listens on standard HTTP or HTTPS ports.
- It registers all requests via HTTP request log.
- It offers a public API via HTTP / HTTPS, secured with credentials and allowed IP address list.
- There is an application specific audit, which records the activity of each user on each system object.
- Each user access to any section of the application can be restricted, and even administrators with restricted permissions can be created.
- The application incorporates a dual authentication system.
- The application incorporates a delegated authentication system (LDAP, AD).
- A read-only system can be built. With no access to device configurations.
- Confidential information (passwords) can be stored encrypted in the database.
- The application connects to the main database using a standard MySQL / TCP connection.
- Part of the code is accessible (OpenSource) and that of the enterprise version can be requested under specific contract conditions (for customers only).
- There is a strong implementation of security policies regarding passwords (length, forced change, history, type of valid characters, etc.)

Possible vulnerabilities and safeguards

- Attacks on the filesystem. The configuration container must be accessed. Solutions:
  1. It is protected in a similar fashion as the server, by means of a secured external NFS system.
- Brute force or dictionary attacks against user authentication. Solution:
  1. Implement a hard password policy (point 14).
  2. Implement a double authentication system (point 8).
- Traffic capture (eavesdropping) of traffic to the console. Solution:
  1. Implement SSL/TLS.
- Traffic capture (eavesdropping) of traffic to the database. Solution:
  1. Implement SSL/TLS.
- SQL injection attacks to obtain confidential information from the application database. Solution:
  1. Implement encrypted data storage.
- Application user misuse (intentional or unintended). Solution:
  1. Activate the audit log and show the users that it exists and its accuracy.
  2. Activate the extended ACL system to restrict the functions of each user as much as possible.
  3. Export the audit log to an external system on a regular basis.

- Execution of malicious code in local console tools, replacing binary files. Solution:
  1. Enforcing server security (hardening) of the server that contains the application.

Agents

- It can be run without superuser permissions (with limited features).
- Remote agent management can be disabled (locally and remotely), so that the impact of a break-in on the central system can be minimized.
- The agent does not listen to network ports, it is the one who connects to Pandora FMS server.
- There is a record of each execution.
- Configuration files are protected by default through file system permissions. Only a user with super administrator permissions can modify them.
- 100% of the code is accessible (under opensource GPL2 licence).

Possible vulnerabilities and safeguards

- Intrusion into the central system that allows distributing malicious command execution to agents. Solutions:
  1. Limit which users can make these policy or configuration modifications (via ordinary console ACL or extended ACL).
  2. Activate the “readonly” mode of the agents (they do not allow configuration remote modifications) for those especially sensitive systems.
- Vulnerability in the filesystem that allows modifying files. Solution:
  1. Correct permission settings.
- Execution of plugins or malicious commands. Solution:
  1. Limit which users can upload executables (via ordinary console ACL or extended ACL).
  2. Perform an audit of new plugins.

Database

- It is a standard product (MySQL)

Possible vulnerabilities and safeguards

- Eavesdropping (network traffic capture). Solution:
  1. Implementation of a secure TLS connection. MySQL supports it.
- Incorrect permissions. Solution:
  1. Correct configuration of access permissions.
- Known MySQL weak spots. It is advisable to establish an update plan for the MySQL server in which you can have it as updated as possible and therefore get rid of any vulnerabilities that old versions may have.

Base system hardening

The hardening or system assurance is a key point in the global security strategy of a company. As manufacturers we issue a series of recommendations to perform a secure installation of all Pandora FMS components, based on a standard RHEL7 platform or its equivalent Centos7. These same recommendations are valid for any other monitoring system based on GNU/Linux.

Access credentials

To access the system, nominative access users will be created, without privileges and with access restricted to the needs they have. Ideally, the authentication of each user should be integrated with a double authentication system, based on tokens. There are free and secure alternatives such as Google Authenticator® integrable on GNU/Linux, although outside the scope of this guide. Seriously consider using them.

If it is necessary to create other users for applications, they should be users without remote access (to do so, disable their shell or equivalent method).

Superuser access

In case certain users need to have administrator permissions, the sudo command is used.

Keep your system up to date

You only need to be connected to the network or configure your yum system to use a proxy server.

This command can cause potential problems with changing libraries, configurations, etc. It is important not to skip the system upgrade, especially when you are putting the system into production. If you are checking an already active production system, you may need to upgrade only those critical components, i.e. those with a vulnerability, for example if you would like to upgrade only mysql server, use the command with the package name you want to upgrade.

```
yum update mysql-server
```

Upgrading the system is a process that should be done periodically. Using the system package inventory, you can check for vulnerable versions and run emergency upgrades.

Access audit

It is necessary to have the security log `/var/log/secure` active and to monitor those logs with the monitoring (which we will see later).

By default CentOS comes with this enabled. If not, check the `/etc/rsyslog.conf` or `/etc/syslog.conf` .

We recommend that you take the logs of the audit system and collect them with an external log management system, Pandora FMS can do it easily and it will be useful to establish alerts or review them in a centralized way in case of need.

Securing the SSH server

The SSH server allows us to connect remotely to our GNU/Linux systems to execute commands, so it is a critical point and must be secured by paying attention to the following points (to do so, edit the file `/etc/ssh/sshd_config` and then restart the service).

- Modify default port (e.g. to 31122)

```
#Port22     ->     Port 31122
```

- Disable root login for superuser

```
#PermitRootLogin yes       ->    PermitRootLogin no
```

- Disable port forwarding

```
#AllowTcpForwarding yes       ->    AllowTcpForwarding no
```

- Disable tunneling

```
#PermitTunnel no       ->    PermitTunnel no
```

- Remove SSH keys for remote root access. We assume that there is only one valid user for remote access (e.g. artica). If there are others, they should also be checked. To do this, we look at the contents of the file `/home/artica/.ssh/authorized_keys` and see which machines we are from. Delete it if we think there should not be any.
- Set a standard remote access warning explaining that the server is private access and that anyone without credentials should disconnect.

```
Banner /etc/issue.net
```

Securing the MySQL server

Listening port. If the MySQL server has to service the outside, we simply check that the root credentials are secure. If MySQL only services an internal element, we make sure that it only listens on localhost:

```
netstat -an | grep 3306 | grep LIST
tcp        0      0 0.0.0.0:3306            0.0.0.0:*               LISTEN
```

In this case it is listening for everyone, we must ensure it. To do this, we will edit the file `/etc/my.cnf` and in the section `[mysqld]` we will add the following line:

```
bind-address = 127.0.0.1
```

And we will verify again that it is listening, but now only on localhost after restarting the service:

```
netstat -an | grep 3306 | grep LIST
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
```

MySQL Password

Connect to the MySQL console with a privileged user:

```
mysql –h host –u root –p
```

Verify that the password is secure and that we have been asked for a password. If not, set it with the command:

```
mysqladmin password
```

This security measure is essential to protect our databases not only against external attacks but also against misuse by internal users.

Securing the Apache web server

We will add the following line to the `/etc/httpd/conf/httpd.conf` file to hide the apache and OS version in the server information headers.

```
ServerTokens Prod
```

PHP Application Engine

In order to secure the application engine on which Pandora FMS works, it could be necessary, in some environments especially sensitive in terms of security, to secure the access to the application so that the session cookies are only transmitted with SSL.

To do this, in the `php.ini` file include the following tokens configuration:

```
session.cookie_httponly = 1
session.cookie_secure = 1
```

This will cause the application to not work when used over
HTTP (without encryption).

Minimization of system services

This technique, which can be very exhaustive, simply consists of eliminating everything that is not necessary in the system. This way we avoid possible problems in the future with misconfigured applications that we do not really need. To simplify the approach to this practice, we will consider only those applications that have an open port on the machine, for this, we will run the following command:

```
netstat -tulpn
```

It should return a result for each listening port, something similar to this, but not the same:

```
Proto Recv-Q Send-Q Local Address           Foreign Address         State       
PID/Program name
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      
996/master
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN      
75171/httpd
tcp        0      0 0.0.0.0:31122           0.0.0.0:*               LISTEN      
872/sshd
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      
75097/mysqld
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      
75171/httpd
tcp        0      0 0.0.0.0:6099            0.0.0.0:*               LISTEN      
7721/Xvfb
tcp6       0      0 :::4444                 :::*                    LISTEN      
7726/java
tcp6       0      0 :::34599                :::*                    LISTEN      
7726/java
tcp6       0      0 :::6099                 :::*                    LISTEN      
7721/Xvfb
```

If we have disabled IPv6 we should not see tcp6 lines unless they are services that were started in ipv6 mode and have been left without restarting after making a change in the system with sysctl.

We should investigate each port and know the application behind it. In this case, 443, 80 seems to be from the `http` service, but to be sure we will analyze which system processes are using each port. To do this we will use the lsof command, which will have to be installed with yum because it is not installed by default.

Those services that listen on localhost (`127.0.0.1`) are more secure than those that listen to all IP addresses (`0.0.0.0`) and some of them, if they are listening in open, we should try to secure them to listen only to localhost. In this example screen it has been done for example for MySQL (`3306`).

For example, we see that the main postfix process is running. As we do not need this service, we uninstall it from the system:

```
yum remove postfix
```

By investigating the PID of each of the processes in doubt (see previous step) we will see the process that is on that port:

```
ps aux | grep 7726 root 7726 0.1 8.5 3258724 248608 ? Sl Mar09 60:01
/usr/bin/java -jar /usr/lib/pwr/selenium-server-standalone-2.53.1.jar -host
185.247.117.28 -port 4444 -firefoxProfileTemplate /opt/firefox_profile root
79041 0.0 0.0 112716 960 pts/4 S+ 11:54 0:00 grep --color=auto 7726
```

And if we are not using that service, we can remove it.

This process of “investigation” of processes must be exhaustive and repetitive over time. By means of the Pandora FMS process inventory system, we should verify that no new processes are started along the time. A listening port in a server is something very significant from the security point of view, it would be like a window in the front of the building. We may believe that it is closed and secure, but a window will always be a possible entry point for a qualified and motivated intruder.

Additional configuration

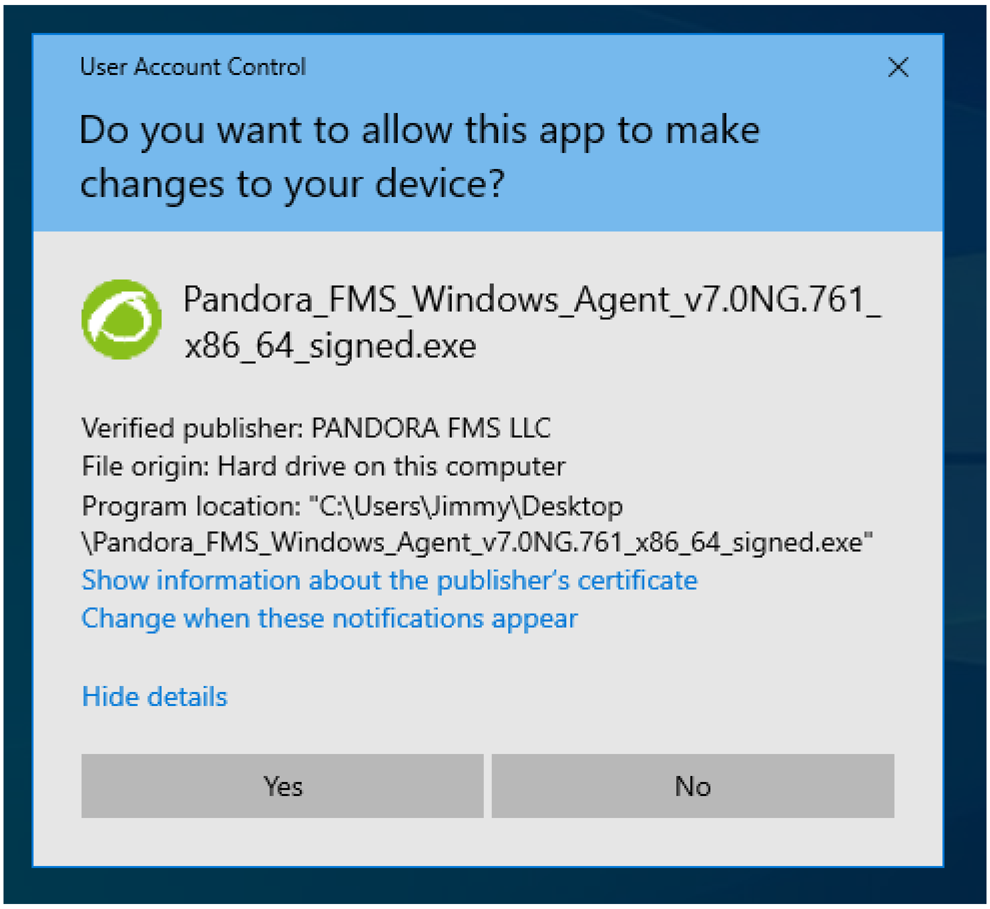
NTP time synchronization

It is recommended to configure the time synchronization of the system:

```
yum install ntpdate
echo "ntpdate 0.us.pool.ntp.org"> /etc/cron.daily/ntp
chmod 755 /etc/cron.daily/ntp
```

Local Monitoring

The system should have a Pandora FMS Software Agent installed and launched against our server. For MS Windows® operating system, from version 761 onwards, the installation executables are signed.



User Account Control

Do you want to allow this app to make changes to your device?

Pandora_FMS_Windows_Agent_v7.0NG.761_x86_64_signed.exe

Verified publisher: PANDORA FMS LLC
File origin: Hard drive on this computer
Program location: "C:\Users\Jimmy\Desktop\Pandora_FMS_Windows_Agent_v7.0NG.761_x86_64_signed.exe"

Show information about the publisher's certificate
Change when these notifications appear
Hide details
Yes
No

The following active checks are recommended in addition to the standard checks:

- Active security plugin.
- Complete system inventory (especially users and installed packages).
- Collection of system and security logs:

```
module_plugin grep_log_module /var/log/messages Syslog \.\*
module_plugin grep_log_module /var/log/secure Secure \.\*
```

Once the Software Agent is installed, at least the following information must be manually defined in the agent tab:

- Description.
- IP address (if you have several, put all of them).
- Group.
- Department, responsible and physical location (custom fields).

Security monitoring in GNU/Linux

The official plugin allows to proactively monitor the security in the agent, in each execution, almost in real time, offering some checks that can alert us of some relevant events in a proactive way.

This plugin is intended to run only on modern GNU/Linux computers. It is prepared to run on 64 and 32 bits.

It contains a custom build of John the ripper 1.8 + Contrib patches with 32 and 64 static binaries. The main concept of the plugin is to be monolithic, detect what can be hardened and try to resolve differences between distros without asking anything to the administrator, so the deployment could be the same for any system, ignoring versions, distro or architecture.

This plugin will check:

- User password audit check, using the dictionary (provided) with the 500 most common passwords. This usually takes no more than a few seconds. If you have hundreds of users, you probably need to customize the plugin run to run only every 2-6 hours. You can customize the password dictionary by simply adding your organization's typical password to the “basic_security/password-list” file.
- Check that SSH does not run on the default port.
- Check that FTP does not run on the default port.
- Check that SSH does not allow root access.
- Check if there is a MySQL running without the root password defined.
- Other checks.

Go back to Pandora FMS documentation index