



# Considerations on Plugins Development



<https://pandorafms.com/manual/!current/>

Permanent link:

[https://pandorafms.com/manual/!current/en/documentation/pandorafms/technical\\_reference/04\\_anexo\\_plugins\\_considerations](https://pandorafms.com/manual/!current/en/documentation/pandorafms/technical_reference/04_anexo_plugins_considerations)

2024/06/10 14:36



# Considerations on Plugins Development

We are working on the translation of the Pandora FMS documentation. Sorry for any inconvenience.

## Introduction

The plugins allows to Pandora FMS to get information that requires a complex process or that requires the use of complex systems or APIs. Examples of plugins could be the Oracle® database monitoring that requires a complex process for the monitoring and also some auto-discovery tasks. Other example could be a simple HTML parse, but that requires some that Goliat ([checking web server](#)) can't do.

## Differences in Implementation and Performance

Pandora FMS offers two possibilities when executing plugins: execution in the [Software agent](#) or in the server.

- The server plugins do independent executions to collect each information piece. The server plugin execution is very difficult so it is only possible for plugins that are “lights”, this is, that does need few or very few queries to get a single piece of information. A server plugin could be an specific HTML parse plugin that requires 2 or 3 queries and so it will charge a little the server.
- The Software agent plugins allow to obtain several modules at the same time and for this reason they are much more flexible than the server plugins. They are perfects for plugins that need several queries to get an information piece; they allow more flexibility to the programmer so it's possible to return several modules at the same time.

## Recon Tasks

To do recon tasks on plugins that need it, we have two possibilities:

The first one consists on using the the Pandora server Recon server. To do this, it will be necessary to create the ad-hoc code for the specific technology or situation. The Recon Tasks loads the Pandora server, so, if for doing the recon task are necessary lot of data requests, this option should't be consider.

It is also possible to create a recon task using an agent plugin. Usually, the agent plugins returns modules that are attached to the XML that the agent sends to the Pandora server. But, consider that when installing the agent in a machine with it, Tentacle is also installed, and this allow to send XML to the Pandora server. To do a recon task from an agent plugin, it is possible to use this, and

besides adding the modules to the agent as a common plugin does, to give our plugin the capacity to send XMLs to Pandora with the information of other agents updated as a recon task would do.

The idea is that the plugin, besides creating the average modules, collects the information and create and send the XML simulating other agents installed if necessary.

The reason to create a plugin that sends data through XML and besides does recon task is to could distribute the monitoring load in different machines and not centralize it in the server.

## Server Plugin or Agent Plugin?

A **server plugin** should be used when:

- The load of each execution is small, for example, simple queries.
- If the Recon Task requires low data process.
- If the Recon Task execution intervals are large, for example, once a week

An **agent plugin** will be used when:

- The information collection requires lot of process or lot of queries.
- The associated Recon Task requires a high process load or lot of queries.
- The Recon Task execution intervals are close to the common execution intervals for agents, for example, every 5 minutes.

## Standardization in Development

In order that all plugins would be the more standard possible, and that they have similar features, you should consider the following aspects:

### Plugin and Extension Versioning

Pandora FMS follows a system of versions for the plugins that has the following format:

```
v1r1
```

Being:

- vX: plugin version, the step of one version to another is made when a new important functionality is added or an error that makes impossible the correct working of the plugin is corrected. The first version is the v1.
- rY: revision of the plugin, the transition from one revision to another occurs when a bug is fixed or a minor functionality is implemented. The first revision is r1.

Always that there would be a change to a new version, should be started by the first revision, that

is, if a plugin is in v1r5 version and for get a higher number of version, then version becoms to v2r1.

## Usage and Plugin version

All plugins should respond to a call without parameters or with a short option type -h or long type -help, showing the command for its execution and the different parameters of it. Besides, it will be necessary to show the version of the plugin. For example:

```
$ ./myplugin
myplugin version: v1r1

Usage myplugin <param1> <param2> <param3>
```

[Go back to Pandora FMS documentation index](#)