PANDORAFMS

Inventory and CMDB

Inventory and CMDB

Back to Pandora FMS documentation index

Inventory

The Inventory functionality provides all the utilities necessary to manage the entire set of objects available to the company, find out their location, their manufacturer, their support contracts, contact persons and telephone numbers, references and serial numbers, etc. Inventory objects can be associated to tickets, allowing efficient management of the company's material and intangible resources.

Within the inventory there are different types of objects,
with their own fields. It is possible to create new types and
modify, add or delete fields of any type of object.

Creating an Inventory Item Type

Inventory item types are used to customize the different fields that the item will have. You can manage the object types by clicking on the CMDB option → Object types.

To edit an object type, click on its name (Name column):

A type of inventory object has associated a name, an icon, the minimum value that it must have in stock (stock) and a description. You can also define whether or not this type will appear as root in the inventory tree view.

Custom Fields

Each type of object has a number of custom fields. For example, the inventory object type named Software has the associated field Version (Version):

To add more fields such as Location (Location) and Manufacturer (Software developer), click on the Create button and fill in the fields as explained in "Properties of custom fields" and press the Create button again.

When there are several custom fields, you can change the relative order between them with the arrow buttons in the actions column (Actions). You can also delete each one in that column by using the trash can icon.

Custom Field Properties

- Label: Label of the object.
- Types: Type of information to fill in later, the fields can be:
    - Numeric (numeric).
    - Checkbox (check box).
    - Text (text).
    - Combo (list of options): When selected, a text box called Combo values will be enabled where you must enter the different values separated by commas (,).
    - External (external field).
    - Date (date).
- Unique: If this option is activated, there can only be one inventory object with the same value for this field (similar to an identifier or ID).
- Global field: If this option is activated, it is allowed to add this custom field to all types of objects. Use with caution.
- Inherit: If this option is activated, the owner, users and companies values of the parent object will be inherited.
- Show in list: If this option is activated, it is shown in the general list by default.
- Prevent updates: If this option is activated, the field will not be updated when receiving remote inventory data.

Fields of external type

The external type fields will show information from additional tables (added manually by an administrator) in the Integria IMS database.

To configure and be able to use an external type field, you need to create a new inventory object with the External Type selected and all the necessary fields for its configuration will appear:

- External table name: Exact name of the external table, in the database, of the table where this

external field is located. Mandatory.
- Field to be displayed: Field to be displayed, name of the column that contains the values that will finally be displayed in the inventory object. Mandatory.

For the creation of an external type being parent of another external object:

- Primary key: Identifier field or "primary key", normally it will correspond to the ID column. It is only necessary if it is a parent table/field.

For the creation of an external type being child of another external object:

- Foreign key: Field of the child table that relates both tables, corresponds to the Foreign Key in the child tableeither. It is only necessary if it is a child table/field.
- Parent table name: Name of the parent table. It is only necessary if it is a child table/field.

With this configuration it will be possible to use an external type field. When creating or modifying an inventory object of a type that contains external fields, a modal window will appear showing the information from the external table:

Also, external type fields can be related to each other, creating dependencies. To do this, the two external tables to be used in our two external fields must also be related by one of their columns, which will usually be an ID. Usually the parent table will have an ID column that will be its `primary key` and the child table will have a column of the ID_relation type that will correspond to its `foreign key`, a column that will relate this child table to its parent table. This will cause the second field to display different values depending on the value you select in the first field.

Case study

You have a manually added table, `t_ubication`, with two columns: `id` and `ubication`, the first is the unique identifier, the second is the useful value to display.

To configure an external type field that makes use of the information contained in this table, configure it as follows:

Thus, firstly the name of the table `t_ubication` is used, and secondly the column `ubication` to be used in the external type field called `Ubication`.

Once this is done, when creating an inventory object of the type that contains this external field, the following steps must be followed:

- Creation of inventory object in the external field:

- Click on the magnifying glass icon and choose a value from the external table:

- After selecting one of the locations shown in the popup dialog (Madrid) it will be selected on the object.

- Now it will be possible to create a second field of external type that will be related to the Location field used. The steps are similar, but completing the additional fields Primary key in the parent field (Location), and Foreign key and Parent table name in the child field (Location).

- You must create a table called t_location, with three columns: id, location and id_ubication.
- Now you must complete the corresponding fields to relate this table with the previous one. In the parent field add the primary key:

- And you must create the child field as follows:

The Primary key fields have been configured in the parent table with its column id, and Foreign key and Parent table name with the foreign key column id_ubication of the child table, and the name of its parent table.

In case you have followed the steps as in the example, you will have created a dependency relationship, and depending on the value you choose in the first custom field, in the second you will have one or the other possibilities.

In this example we have chosen Madrid in the Location field and Atocha will appear as the only eligible option as it is the only one that exists with the id_ubication equal to 1:

The maintenance of these external tables is done manually
by operating with them by means of SQL statements, or
else, using the external table editor provided by Integria
IMS.

Editing external tables

In the list of inventory object types, those that have external fields will have an icon for editing tables.

On these tables it is possible to carry out editing and deletion operations of existing data, as well as adding new data.

Creating an Inventory Item

From the left side menu it is accessed from CMDB → Create new object.

The most important fields in this form are:

- Owner (Owner).
- Associated companies.
- Associated users.
- Public (Public).

These fields define who can see the object: In this way the object will be accessible by the ownerio, users associated directly with the object, or users belonging to an associated company. Also, if the Public flag is checked, all users will be able to see said object.

An inventory item can have a "hierarchy" of items (one item can be a child of another). This is defined by choosing a parent. In addition to this parent-child relationship, relationships between objects can be established.

Integria IMS inventory has a stock control system. To manage stock, all inventory objects have a "status" field that allows you to keep a stock system (the date of receipt and removal can also be recorded). The possible states are:

- New.
- In use.
- Unused.
- Issued.
- In addition to all those defined by the user in the inventory configuration menu of Pandora FMS.

Example:

The relationships tab shows the other objects related to this one. New objects can be associated, just choose from the list and link as shown below. In this way, other objects can be associated besides the parent object.

The contacts tab shows the people associated with that inventory item. By clicking on any of them you can access your information directly.

The tickets tab associated with the inventory object indicates which tickets this object has suffered, and in what state they are. You can click on any of them to access your information directly.

The tracking tab (Tracking) shows the changes that occur in the inventory object such as the change of owner, if an object type is associated, a parent object, and so on.

TRACKING

Inventory object #31: ❓

Description	User	Date
Updated	Default Admin	2021-11-08 11:55:36
Owner changed -> Old: 'Default Admin' - New: 'Jaime Blanco'	Default Admin	2021-11-08 11:55:36
Inventory object type added -> Software	Default Admin	2021-11-08 11:55:36
Parent changed: -> Old: 'sample_linux_host' - New: 'sample_linux_host'	Default Admin	2021-11-08 11:55:36
Parent added -> sample_linux_host	Default Admin	2021-11-08 11:55:36
Users updated	Default Admin	2021-11-08 11:55:36
Companies updated	Default Admin	2021-11-08 11:55:36
Updated	Default Admin	2021-01-21 19:45:42
Inventory object type added -> Software	Default Admin	2021-01-21 19:45:42

Inventory Searches

The fields that can be searched are:

- Search: Free text field that refers to the name or description of the inventory if the token is active. This is true if no custom fields are selected. In this case, the search also limits the value of these fields.
- Search for description : Add the description of the object to the Search field.
- Status: Status of the object.
- Block size per pagination: Select the size of the pagination block. By default, this is configured in general options and is limited between 2 and 1000.
- Associated company: Associated company
- Contract: Contract.
- Parent object: Parent object.
- Owner: Owner.
- Associated user: Associated user.
- Object Type. It is possible to select more than one type of object in the search using the selector
- Last updated: last update.
- Object type: Type of object.

❌

You can also export search results in CSV format. The inventory search result can be displayed in two modes: tree or list.

The tree view shows inventory items grouped by type. The stock is also displayed, showing:

- Total units.
- New.
- Without using.
- Minimum stock. If the number of total units is less than the minimum stock, it will be marked in red.

In the list mode view, information about the inventory is displayed, as well as its custom fields that have been marked to display.

.

Import inventory data from CSV

To do this, the CSV file is loaded with a header and a line for each inventory object. The header must be in the first row of our CSV and must contain at least the name column since the name of the inventory object must exist.

- name: Required. The setup option that allows you to have duplicate inventory object names will be taken into account.
- owner (Owner): This column collects the identifiers of the users who own the object, which must exist in the database. If this column is not entered, the value entered by default will be empty.
- id_parent (Parent Object): In this column will go the identifier of the parent inventory object, the cuals must exist in the database. Default is zero (0).
- id_manufacturer (Company): In this column will go the identifier of the companies to which each inventory object belongs, with which these companies must exist in the database. Default is zero (0).
- id_contract (Contract): This column will contain the identifier of the contract to which each inventory object belongs. Default is zero (0).
- status (Status): In this column you can collect the four types of status that Integria IMS defines by default (new, inuse, unused, issued) or a user-defined state.
- receipt_date (Receipt Date): The date of receipt will go in this column and it must have a valid date format.
- issue_date (Date of use): In this column the date of use will go and it must have a valid date format.
- description
- public: It will only admit value 0 or 1 depending on whether you want it to be public or not. Default is one (1).
- id_company (Associated companies) the associated companies can be one or several, therefore we will introduce the ids as follows: [id1 .id2 .id3 .id4]. It is important to use brackets and within them use points to separate the elements.
- associated_user (Associated users): Associated users can be one or more, therefore we will introduce the ids as follows: [id1 . id2 .id3 . id4] . It is important to use brackets and within them use points to separate the elements.
- id_object_type (Object Type): In this column will go the identifier of the types of objects which must exist in the database. Default is zero (0).
- The rest of the columns that are added will correspond to the custom fields of a type of object, there are four different types of fields which must respect the limitations of said fields:
  - Text type fields: They do not have any limitations.

- Numeric type fields: They only allow numbers, therefore the column of this field must contain only numbers.
- Combo type fields: They allow a list of words, therefore the characters entered in this column can only be those that are included in said list.
- External type fields: These fields allow you to associate external tables to enter the values, it will be validated that said table and its references exist.

All custom fields will also need to match whether they are unique or not. This depends on whether you have selected to create custom fields, which will result in not being able to enter values that already exist in these custom fields.

Example of CSV file edited in a spreadsheet:

Remote inventory based on Pandora FMS agents

This Enterprise functionality allows you to pull custom inventory information from any device and operating system.

The remote inventory is based on the use of Pandora FMS software agents that are installed on the machines from which the information is to be extracted. The agents collect inventory information through scripts and send the data to the server where Integria IMS is running. Once the information is received, IIMS will process it and create the relevant inventory objects based on the data received.

Unlike the Inventory Synchronization with Pandora FMS, to use the Remote Inventory based on Pandora FMS agents it is not necessary to install the Pandora FMS application, it is only necessary to install the agents on the machines you want.

Internals of Remote Inventory

The Pandora FMS agents send an XML file with the inventory information to the server that supports Integria IMS. IIMS processes these files through the maintenance script: file processing is typically done every 5 minutes. These XML should be received in the `/attachment/inventory` directory located inside the IIMS directory.

With the inventory information sent in the XML several inventory objects will be created. In such a way that if an agent sends information about 12 applications installed on the system, 13 inventory objects will be created (one for the agent and 12 for the different installed applications).

When information about a device or inventory item is received, two things can happen:

1. That the item exists: Then the inventory information will be updated.
2. That the item does not exist: In this case, a new inventory object will be created only if the typeor associated object is defined in Integria.

It's important to define the object types you'll be using before you begin inventorying, the following explains how data acquisition works in detail, and you can see how types are associated with different items.

Pandora FMS agents are associated with the type called Pandora agents, so for the inventory to work it must be defined. The other inventory objects are associated to their types as explained below.

XML example:

```
<?xml version='1.0' encoding='UTF-8'?>
<agent_data description='' group='' os_name='linux' os_version='"CentOS release
6.4
(Final)"' interval='300' version='7.0NG' timestamp='2021/10/22 09:27:38'
agent_name='sample_linux_host' timezone_offset='0' address='192.168.70.163'>
  <inventory>
   <inventory_module>
     <name><![CDATA[Users]]></name>
     <datalist>
       <data><![CDATA[root]]></data>
     </datalist>
   </inventory_module>
   <inventory_module>
     <name><![CDATA[Process]]></name>
```

```
        <datalist>
          <data><![CDATA[COMMAND]]></data>
          <data><![CDATA[/sbin/init]]></data>
          <data><![CDATA[[kthreadd|]]]></data>
          <data><![CDATA[[migration/0|]]]></data>
          <data><![CDATA[[ksoftirqd/0|]]]></data>
          <data><![CDATA[[scsi_eh_6|]]]></data>
          <data><![CDATA[sh -c ps -eo command | tr -d "" ]]></data>
          <data><![CDATA[ps -eo command]]></data>
          <data><![CDATA[tr -d ]]></data>
        </datalist>
      </inventory_module>
      <inventory_module>
        <name><![CDATA[Software]]></name>
        <datalist>
          <data><![CDATA[acl;2.2.49;Access control list utilities]]></data>
          <data><![CDATA[acpid;1.0.10;ACPI Event Daemon]]></data>
          <data><![CDATA[alsa-lib;1.0.22;The Advanced Linux Sound Architecture
(ALSA)
          library]]></data>
          <data><![CDATA[alsa-plugins-pulseaudio;1.0.21;Alsa to PulseAudio
backend]]>
          </data>
          <data><![CDATA[zenity;2.28.0;Display dialog boxes from shell
scripts]]></data>
          <data><![CDATA[zip;3.0;A file compression and packaging utility
compatible
          with PKZIP]]></data>
          <data><![CDATA[zlib;1.2.3;The zlib compression and decompression
library]]>
          </data>
        </datalist>
      </inventory_module>
      <inventory_module>
        <name><![CDATA[File system]]></name>
        <datalist>
          <data><![CDATA[tmpfs;0;244M;/dev/shm]]></data>
          <data><![CDATA[/dev/sda1;34M;426M;/boot]]></data>
        </datalist>
      </inventory_module>
    </inventory>
    </agent_data>
```

IIMS will create an item for the agent that reported the XML and then process the different inventory modules separately, for example:

```
  <inventory_module>
    <name><![CDATA[Software]]></name>
    <datalist>
      <data><![CDATA[acl;2.2.49;Access control list utilities]]></data>
      <data><![CDATA[acpid;1.0.10;ACPI Event Daemon]]></data>
```

```
      <data><![CDATA[alsa-lib;1.0.22;The Advanced Linux Sound Architecture
(ALSA)
      library]]></data>
      <data><![CDATA[alsa-plugins-pulseaudio;1.0.21;Alsa to PulseAudio
backend]]>
      </data>
      <data><![CDATA[zenity;2.28.0;Display dialog boxes from shell
scripts]]></data>
      <data><![CDATA[zip;3.0;A file compression and packaging utility
compatible
      with PKZIP]]></data>
      <data><![CDATA[zlib;1.2.3;The zlib compression and decompression
library]]>
      </data>
    </datalist>
  </inventory_module>
```

The name of the inventory module defined in the XML will be chosen, in this case Software, to identify the type of object associated with these elements.

Then, inventory items with that type will be created for each of the items reported in the XML:

```
<datalist>
   <data><![CDATA[acl;2.2.49;Access control list utilities]]></data>
   <data><![CDATA[acpid;1.0.10;ACPI Event Daemon]]></data>
   <data><![CDATA[alsa-lib;1.0.22;The Advanced Linux Sound Architecture (ALSA)
   library]]></data>
   <data><![CDATA[alsa-plugins-pulseaudio;1.0.21;Alsa to PulseAudio backend]]>
   </data>
   <data><![CDATA[zenity;2.28.0;Display dialog boxes from shell
scripts]]></data>
   <data><![CDATA[zip;3.0;A file compression and packaging utility compatible
   with PKZIP]]></data>
   <data><![CDATA[zlib;1.2.3;The zlib compression and decompression library]]>
   </data>
</datalist>
```

Inventory items separate information by character ; . The information would be broken down as can be seen below:

```
<data><![CDATA[acl;2.2.49;Access control list utilities]]></data>

acl 2.2.49 Access control list utilities
```

The first component defines the name for the new inventory object. The rest of the components will be associated with the object type fields in the same order in which they appear in the inventory object definition. In this case the second element is associated with Version and the third with Description.

Furthermore, all the objects corresponding to the elements tThe agent that reported them through the XML will be entered as parent. The end result for the inventory object created for that XML element will look like this image:

In the inventory, along with all the objects that have been created based on the various inventory modules in the XML, an item will appear for the agent, in this case called `localhost.localdomain`.

By default, Integria IMS has the main inventory modules predefined for MS Windows® agents, such as Software, Patches, HD, Video, CPU, Services and RAM:

Remote Inventory Settings

The automatic inventory configuration can be found in the Setup menu in the Pandora FMS Inventory tab, Remote Inventory section.

The configurable fields are:

- Default owner: Default owner of the new inventory item.
- Associated company: Default companies for the new inventory item.
- Associated user: Default users for the new inventory item.

The Default owner field is required, if it is not set the inventory objects will not be created. The other fields serve for the purposes of configuring access ACLs and viewing objects.

In addition to the configuration of the mentioned fields, it is necessary to enable the maintenance script. You can see how to install this script in the "Installing the maintenance script" section.

If you do not enable the maintenance script, the files sent by the Pandora FMS agents will not be processed.

The sending of files is done through a TCP connection to port 41121. Therefore, for the correct operation of this functionality, you may have to check the routing rules or network firewalls.

Configuration of Pandora FMS agents

To extract the information from the devices you must install Pandora FMS agents. You can see the compatibility of the Pandora FMS agent with your systems in the "Software agent requirements of the Pandora FMS manual" section.

The additional parameters to configure in the agent are the following:

- `server_ip`: IP address of the server running Integria IMS.
- `server_path`: Directory to receive files from the server.
- `module_plugin`: Directive to run scripts to extract inventory information.

By default, Pandora FMS agents have a plugin module to extract inventory information:

```
module_plugin inventory 1 cpu ram video nic hd cdrom software
init_services filesystem users process ip route
```

Inventory Customization

If you want to customize the information collected you can modify this script or create your own.

In the section "Creation of local inventory modules" of the Pandora FMS manual describes the XML structure of the inventory modules and examples of how to create inventory modules for GNU/Linux® and MS Windows® platforms.

Back to Pandora FMS documentation index