PANDORAFMS
ENTERPRISE

# Pandora FMS
## Administrator Manual
### Monitoring with IPTraf

*Administrator Manual Monitoring with IPTraf*



# Index

# 1 Changelog

| Date | Author | Change | Version |
|---|---|---|---|
| 10/07/12 | Dario | First Version | v1r1 |

# 2 Introduction

Pandora FMS allows you to monitor network traffic statistics processed by **IPTraf**.

**IPTraf** collects network activity statistics from one or all interfaces and stores all information in a logfile.

A **passive collector** filters the information based on rules and creates a tree structure with all the information. One XML file per IP detected will be generated using the network activity information contained in the tree structure.

Once XML files are processed, one agent per IP detected will appear in Pandora FMS, these agents will have several modules with their network traffic information.

# 3 Compatibility Matrix

| | |
|---|---|
| **Was tested in these systems** | • IPTraf 1.1.1 |

| | |
|---|---|
| **It should work in these systems** | • IPTraf 1.1.1 and higher |

# 4 Documentation provided by the requesting area

These parameters must be provided by the area which request the monitoring services:

- Full path of IPTraf logfile
- Filtering rules for logfile (explained below)

## 4.1. Filtering rules

To understand filtering rules first we must understand IPTraf logfile structure.

### *4.1.1. IPTraf logfile structure*

And example of a log line is:

```
Mon Nov 22 15:41:59 2010; TCP; eth0; 52 bytes; from 192.168.50.2:54879 to
91.121.0.208:80; first packet
```

After the date and hour record there is the protocol, the interface name, the number of bytes transfered, the source ip and port and the destination ip and port. After will appear some information in this case indicates this communication is the first package.

Important data in this line are the interface name, the number of bytes transfered, the source IP and port and the destination IP and port.

### *4.1.2. Collector filtering rules*

The rules have the following structure:

```
[process/discard] [!][ip_src/ip_dst] ip/mask [!][port_src/port_dst] port [!]
[protocol] protocol
```

The **first parameter** could be *process* if you want to process the records which match this rule or *discard* if you want to discard the record that match.

The **second parameter** set the match with source (ip_src) or destination IP (ip_dst). This parameter could be denied with the character (!) before, indicating we want the records that DON'T match with this IP.

The **third parameter** is an IP following by a network mask. If you want only an IP you can set the IP without mask or the mask 32. If a mask is specified all IP in the range will be considered.

For example, 192.168.50.0/24 will be IPs in range 192.168.50.1-192.168.50.254. Otherwise 192.168.50.23 y 192.168.50.23/32 are the same IP 192.168.50.23.

The **fourth parameter** is similar to second one but this time the instead of IP we will filter by source (port_src) or destination port (port_dst). Also it is possible to use character (!) before port to

denied these ports.

The **fifht parameter** are the port numbers which will be used to match the record.

You can specify the following parameters:

- One port with a number. For example 8080.
- An interval separated by a dash character. For example 21-34 to match all ports from 21 to 34 both included.
- A port enumeration separated by comma. For example 21,23,80,8080.
- A combination of intervals and enumerations. For example 21-34,80,8080,43234-43244.

The **sixth parameter** is the protocol used to perform the communication. This parameter could be denied with character (!) before, it indicates you want the records which DON'T match with this protocol. es el protocolo por el que se realiza la comunicacion. Este parametro puede ir negado con el caracter de exclamacion (!) delante, indicando que queremos los registros que NO coincidan con ese protocolo.

You can use the following formats:

- A protocol. For example TCP.
- Several protocol separated by comma. For example TCP,UDP,FTP.
- A special word "all" to match all protocols.

#### *4.1.2.1. Examples*

Some valid rules are:

```
discard src_ip 192.168.70.222/32 !port_dst 21-23,80,8080 protocol all
process src_ip 192.168.70.0/24 !port_src 0 !protocol TCP
process src_ip 192.168.80.0/24 !port_dst 80,8080 protocol UDP,TCP
```

These rules will process the following records:

- All records with source IP an IP in network 192.168.80.X while don't have 80 or 8080 as destination port and use TCP or UDP protocols.
- All records with source IP in network 192.168.70.X with any source port that don't use TCP protocol except record discard by the first rule. The first rule discard record with source IP 192.168.70.222 and destination port different from 21,22,23,80 and 8080 using any protocol.

# 5 Modules of this plugin

The plugin creates modules dinamically based on filtering rules defined and on network traffic detected by IPTraf.

Inside Pandora FMS will appear one agent per IP detected with several modules showing the network statistics for this IP.

# 6 How it works

The *passive collector* is a script called **passive.pl**. This script parses the information and generates XML files in an asyncronous mode. So you must execute th script every time you want to update traffic monitoring information in Pandora FMS.

The script must be executed with root privileges

Before the script is executed IPTraf process must be stopped. After the script execution the log file used must be deleted and IPTraf process must be restarted

In execution command the configuration file is passed as parameter, like this:

```
# ./passive.pl /home/usuario/iptraf/passive.collector.conf
```

The steps to execute the scripts are the following:

1. Stop IPTraf
2. Run passive collector
3. Delete logfile
4. Start IPTraf

The actions performed by the scripts are the following:

1. Parse logfile
2. Apply discard rules
3. Apply process rules
4. Create tree with all information
5. Generate XML files and store in data_in folder of Pandora FMS
6. End execution

# 7 Configuration

The configuration file called passive.collector.conf has the following parameters:

- **incomingdir**: Full path of data_in folder of Pandora.
- **interval**: Interval (in seconds) of script execution. This parameter doesn't mean the scrip will be executed each interval, it is used to set the module and agent intervals in Pandora FMS and allow you to know when the agents and modules are in unknown status. The script execution time is controlled externaly.
- **iface**: Interface name which is listening for network traffic.
- **min_size**: Its possible to filter records based on a minimum size, disabled with value 0.
- **log_path**: Full path of IPTraf logfile.
- **rules**: There are two kind of rules:

1. *discard* : Discard rules are executed first and discard the records that match these rules.
2. *process* : Process rules are executed in second place and filter the remaining records which will be included in the tree.

# 8 Data generated

The data generated by **passive collecter** are XML files. An XML file per IP detected that match the rules is generated. These files are copied in the path defined in parameter *incomingdir* of configuration file which must be the path to data_in folder of Pandora.

The XML content are modules of Pandora FMS which have the network statistics for this IP. An example of XML file could be like this:

```
<agent_data interval='300' os_name='Network' os_vesion='4.0.2' version='N/A'
timestamp='AUTO'
address='192.168.70.1' agent_name='IP_192.168.70.1'>
 <module>
  <name>Port_67</name>
  <type>async_data</type>
  <description>Total bytes of port 67</description>
  <interval>300</interval>
  <data>1312</data>
 </module>
 <module>
  <name>Port_67_Protocol_UDP</name>
  <type>async_data</type>
  <description>Total bytes of port 67 for protocol UDP</description>
  <interval>300</interval>
  <data>1312</data>
 </module>
 <module>
  <name>IP_192.168.70.141</name>
  <type>async_data</type>
  <description>Total bytes of IP 192.168.70.141</description>
  <interval>300</interval>
  <data>1312</data>
 </module>
 <module>
  <name>IP_192.168.70.141_Port_67</name>
  <type>async_data</type>
  <description>Total bytes of IP 192.168.70.141 for port 67</description>
  <interval>300</interval>
  <data>1312</data>
 </module>
 <module>
  <name>Protocol_UDP</name>
  <type>async_data</type>
  <description>Total bytes of Protocol UDP</description>
  <interval>300</interval>
  <data>1312</data>
 </module>
</agent_data>
```