1.1 Installing the framework

1.1.1 Installing the Agents

On each host where there is tools we want to use, we need to install an Agent.

The installation require :

- The log4cpp library and source (for compilation). You can download it on log4cpp.sourceforge.net

1. Uncompress the INMF tar.gz file : `tar -xvzf  INMF-0.1.tar.gz`
2. Go in the INMF/agent directory
3. In the agent directory : `./configure`
4. If there is no error, you can compile the agent : `make`
5. Install the agent : `make install`
6. Create a directory `/etc/INMF`
7. Copy the config file `INMF/contrib/agent.cfg` in this directory

You can find more detailed informations in the INSTALL file.

1.1.2 Configuration of the Agents

The default configuration file is in /etc/inmf/agent.cfg

This file is commented to allow you to make your own configuration.

1.1.3 Installing the Manager

On one host who can accede the entire managed network, you need to install the manager.

The installation require :

- The log4cpp library and source (for compilation). You can download it on log4cpp.sourceforge.net
- MySQL server
- The MySQL interface for C++, you can download it on dev.mysql.com
- An apache server with PHP

1. Uncompress the INMF tar.gz file : `tar -xvzf  INMF-0.1.tar.gz`
2. Go in the INMF/manager directory
3. Change the rule file : `rules.c`
4. In the manager directory : `./configure`

5. If there is no error, you can compile the manger : `make`
6. Install the manager : `make install`
7. Create a directory `/etc/INMF`
8. Copy the config file `INMF/contrib/manager.cfg` in this directory

You can find more detailed informations in the INSTALL file.

1.1.4 Configuration of the Manager

The default configuration file is in /etc/inmf/manager.cfg

This file is commented to allow you to make your own configuration.

The rule file is in src/rules.c

To write the rules you need to have exactly :

- The name of the agent as define in the configuration file. This is to avoid using directly the agent IP in the rules.
- The installed tools on each agent, and there name (referring to the Agents config file).
- The name of the initial alert you can get (see plugins documentation).
- The name and the signification of the data each agent can send (see plugins specification).

The rules language is actually C++. Some functions are provided :

- string value(string DataName) : Return the corresponding value of the data with name DataName. Return a empty string if the data doesn't exist.
- alert(int priority) : Send an alert to the user. Actually write this alert in the MySQL database. The priority is between 1 (low) and 10 (hight).
- int getUserPrio(string user name) : Look in the security policy database the priority affected to this user. The priority is between 1 and 5. If the user is not in the database, the priority will be 0 by default.
- int getNetworkPrio(string IP) : Look in the security policy database the priority affected to the network containing this IP. The priority is between 1 and 5. If the IP doesn't belong to any network in the database, the priority will be 0 by default.
- int getIPPrio(string IP) : Look in the security policy database the priority affected to this IP. The priority is between 1 and 5. If the IP have no entry in the database, the priority will be 0 by default.
- int getTimePrio(string myTime) : Look in the security policy database the priority affected to the Time. The priority is between 1 and 5. Doesn' belong to any time interval in the database, the priority will be 0 by default.
- logger.info(string text) : To write informations in the log file.

To define the decision we have to give value to :

- addDec(string agent, sting tool, string state, string params) :

    - agent : The name of the concerned Agent.
    - tool : The name of the concerned tool on the Agent.
    - state : “run” or “stop” for running or stopping the tool.
    - params : The parameters we want to give to the command for running the tool. Usually empty for stopping tool.
- resetHistory() : reset the list containing all the decision take. We have to use it when we finish one decision process to state a new decision process.
- stopAllHistory : stop all the previously launched tool. We have to use it when we finish one decision process to state a new decision process.

Using this we can construct a decision tree, mainly using the C++ if(condition) {} function. The files in the distribution can be use as an example.

1.1.5 Installation of the web interface

The installation require :

- A HTTP server (like apache) with php.
- It must be install on a server who can accede to the MySQL database of the Manager.

1. Uncompress the INMF tar.gz file : `tar -xvzf INMF-0.1.tar.gz`
2. Put all the files of the `INMF/web` in a directory where the HTTP server can accede it.
3. Configure your HTTP server.
4. Edit the `db.php` file and set the MySQL server address, the user and password and the database name, in the start of the file.

1.2 Installation of the tools

1.2.1 MRTG (Multi Router Traffic Grapher)

We use the latest version of MRTG for monitoring various hosts. At the time of writing this document, the latest stable version is 2.10.14. A patch is required to enable MRTG for creating the RRD archives compatible with Aberrant Behaviour Detection (Using RRDTool 1.1.0).

The patch is available in the tools_proto0.tar.gz file, in the contrib/mrtg.diff file. Think you to the OSSIM project who make this patch.

- Download the MRTG sources at : people.ee.ethz.ch/~oetiker/webtools/mrtg/pub/
- Decompress it.
- Go in the mrtg/bin directory.
- Enter patch -p0 < pathToPatchFile/mrtg.diff
- Follow the normal mrtg installation guide.

You also have to install the development version of RRDTool (1.1.0). You can find it on : people.ee.ethz.ch/~oetiker/webtools/rrdtool

Here are examples on how to configure MRTG :

➢ Monitoring bandwidth: MRTG monitors the "ifInOctets" and "ifOutOctets" using SNMP requests. The configuration file for doing so is provided.

```
WorkDir: /path/to/the/directory
Options[_]: growright
RunAsDaemon: Yes
Interval: 5 #it means 5 minutes here , can be changed
LogFormat: rrdtool
PathAdd: /path/to/rrdtool/bin/
LibAdd: /path/to/rrdtol/lib/perl/
Target[Statistics]: interface:community string@host
MaxBytes[Statistics]: 12500000
```

Of course, one can add to configuration file for drawing graphs with mrtg. Here the data is directed to RRD database. Graphs can be drawn using RRDtool' s graph function.

➢ Monitoring Squid : MRTG monitors various Squid SNMP variables. Out of them all, cacheServerRequests determine the number of requests.

```
            WorkDir: /path/to/the/directory
            Options[_]: growright
            RunAsDaemon: Yes
            Interval: 5 #it means 5 minutes here , can be changed
            LogFormat: rrdtool
            PathAdd: /path/to/rrdtool/bin/
            LibAdd: /path/to/rrdtol/lib/perl/
            Target[cacheServerRequests]:
cacheServerRequests&cacheServerRequests:community@host:3401
            MaxBytes[cacheServerRequests]: 10000000
            Options[cacheServerRequests]: nopercent
```

1.2.2 Qmailmrtg

Qmailmrtg allow us to monitor QMail logs using MRTG. Using it, we can monitor the number of sending mails.

Qmailmrtg require the installation of Multilog (part of the daemontool package : http://cr.yp.to/daemontools.html)

Install qmailmrtg (http://www.inter7.com/index.php?page=qmailmrtg7) using the documentation.

In the configuration file , the following lines need to be added to enable RRD storage.

```
LogFormat: rrdtool
PathAdd: /path/to/rrdtool/bin/
LibAdd: /path/to/rrdtol/lib/perl/
```

1.2.3 TCPTrack

The original TCPTrack software is only showing data on screen. We have to read those data. For this, we have to modify the original software. The next version of TCPTrack will include functions to dump data in one file. Our modified version will run for 30 seconds and print on the standard output the data.

The patch is on the INMF/contrib directory of the project.

1.2.4 Scripts

We are providing several scripts to get informations from log files :

- squiduser.pl : This script is a parser for squid (proxy server) logs. It take an IP and an interval of time (StartTime, StopTime) in parameter and will find the login of the user using this IP in this interval of time. It will store the output in a file called squiduserlog in the format : `User:userid`

  The associated plugin for getting information from this script is plugin_squiduser.pl

  The path to the squid log file has to be set at the beginning of the file.

  This script is used in the abnormal bandwidth exceed scenario. When we have the IP and the port responsible of this exceed, we can get is login name on the proxy server, if the port correspond to a traffic who go thought the proxy server.

- squidcnx.pl : This script is a parser for squid (proxy server) logs. It take an interval of time (StartTime, StopTime) in parameter and will find the login and IP of the user making the maximum number of connection to the server. It will store the output in a file called squidcnxlog in the format : `User:userid\n IP:ip\n Usage:num of cnx\n`

  The associated plugin for getting information from this script is plugin_squidcnx.pl

  The path to the squid log file has to be set at the beginning of the file.

  This script is used in the abnormal proxy connection scenario. When know that there is an abnormal number of connection on the proxy server, we want to find who is responsible of it.

1.2.5 Plugins

- plugin_squiduser.pl : This script will read the dump file of the squiduser script, get the informations (user login) and reset the file. The return is :

```
User:login\n
```

- plugin_squidcnx_rrd.pl : This plugin is collecting data from RRD with aberrant behaviour detection. It is looking for abnormal number of connexion to the squid proxy server sent alarms in the database. If it detect an alarm, it return :

```
Alert:squidcnx\n
FirstFailureTime:Time\n
LastFailureTime:Time\n
```

- plugin_squidcnx.pl : This plugin will read the dump file of the squidcnx script, get the informations (user login and IP) and reset the file. The return is :

```
User:login\n
IP:ip\n
Usage:number of connexion\n
```

- plugin_bandwidth_rrd.pl : This plugin is collecting data from RRD with aberrant behaviour detection. It is looking for bandwidth exceed alarms in the database. If it detect an alarm, it return :

```
Alert:InBw\n if the alert is on the In bandwidth
Alert:OutBw\n if the alert is on the Out bandwidth
FirstFailureTime:Time\n
LastFailureTime:Time\n
```

- plugin_qmailsent_rrd.pl : This plugin is collecting data from RRD with aberrant behaviour detection. It is looking for abnormal number of mail sent alarms in the database. If it detect an alarm, it return :

```
Alert:qmailsent\n
FirstFailureTime:Time\n
LastFailureTime:Time\n
```

- plugin_tcptrack.pl : This plugin is analysing TCPTrack dump file and file get the IP and the port which is using the biggest bandwidth. It return :

```
IP:ip\n
Port:port\n
```