Мониторинг с помощью программных агентов Software Agents

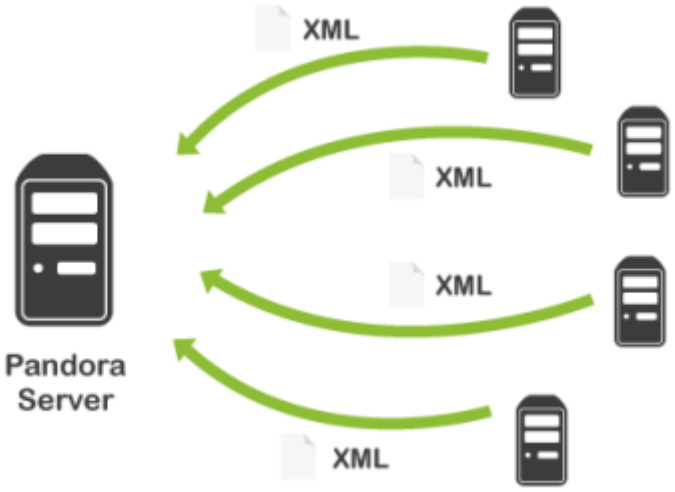
19-03-2024

Мониторинг с помощью программных агентов Software Agents

Вернуться в оглавление Pandora FMS

Мониторинг с помощью программных агентов Software Agents

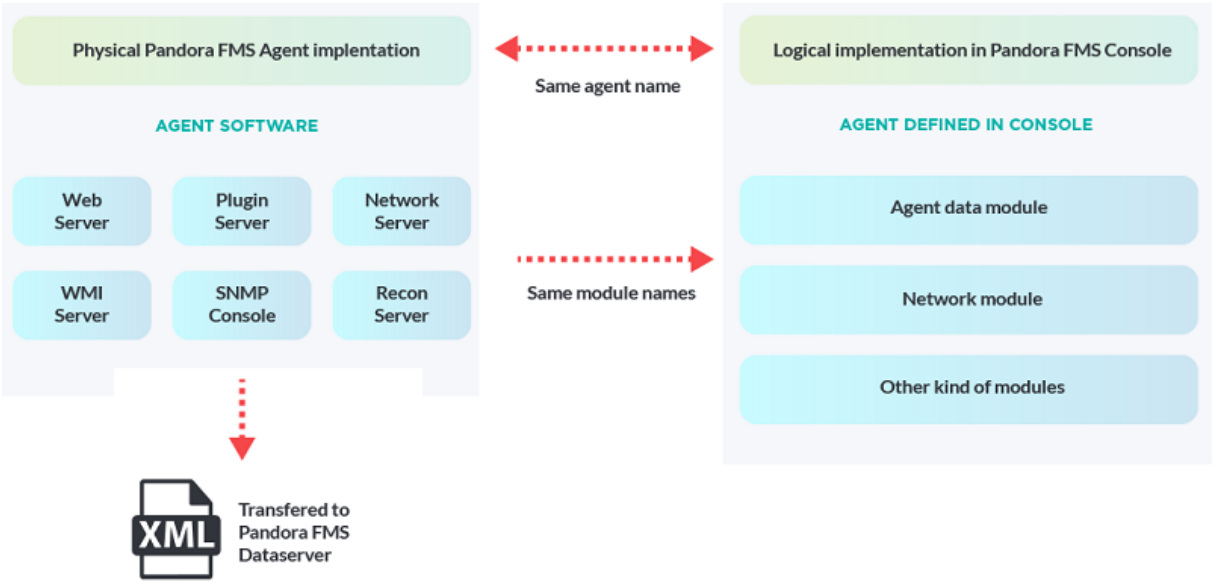
Мониторинг с помощью программных агентов Software Agents



Программные Агенты работают в операционных системах, из которых они собирают информацию. **Каждая проверка**, производимая с системой,такая как использование CPU, свободная память или дисковое пространство, соответствует одному **модулю**. Таким образом, для каждого модуля при каждом выполнении собирается только один элемент информации.

Собственные *директивы* Программного Агента используются для сбора определенных данных непосредственно из операционной системы (например, использование процессора, памяти, события и т.д.), выполняя **собственные команды операционной системы**, следуя инструкциям из заранее предустановленных *скриптов*. Также можно выполнять эти команды напрямую, как и любые другие программы, если они возвращают данные стандартным образом.

Сервер Данных Pandora FMS обрабатывает и сохраняет в базе данных всю информацию, генерируемую программными агентами, которые отправляют ему свои данные в XML-файле.



Логическая схема физического агента/агентов.

Если вы используете версии ранее 7 NG, см. раздел об именовании программных агентов в конце этой статьи.

Конфигурация программного агента

Все конфигурации и параметры хранятся в файле *pandora_agent.conf*, который также устанавливается локально с вашим Программным Агентом. Базовая конфигурация рассматривается в разделе "Конфигурация агентов Pandora FMS", далее раскрывается расширенная конфигурация.

Локальная конфигурация

В конфигурационном файле Программного Агента модули определяются со следующей основной текстовой структурой:

```
module_begin
module_name your module name
module_type generic_data
module_exec your command
module_description your description
```

```
module_end
```

Пример 1

```
module_begin
module_name Files in var spool
module_type generic_data
module_exec ls /var/spool | wc -l
module_description Number of files incoming dir
module_end
```

В среде *nix команда **ls** перечисляет файлы одного каталога и выполняется со строкой *module_exec*, чтобы передать значение команде **wc**, которая подсчитывает количество слов, полученных для того же количества файлов. Значение, возвращаемое этим последним выполнением, является данными, которые будут получены модулем и отображены в мониторинге.

Пример 2

```
module_exec vmstat 1 2 | tail -1 | awk '{ print $13 }'
```

Команда **vmstat** сообщает статистику виртуальной памяти, в данном примере приведены две дополнительные команды для «уточнения» нужной информации. Рекомендуется сначала запустить команду вручную и проанализировать вывод.

```
$> vmstat 1 2 | tail -1 | awk '{ print $13 }'
```

Если результат удовлетворяет требованию, он может быть добавлен в файл конфигурации; впоследствии значение, возвращенное в результате выполнения через Software Agent, будет сохранено в XML как данные модуля.

Пример 3

Любая команда или программа может быть выполнена через *module_exec* при условии, что вывод совместим со значениями, принимаемыми Pandora FMS (числовые, буквенно-цифровые или *булевые*), поэтому можно указывать пользовательские *скрипты*:

```
module_exec myScript.pl --h 127.0.0.1 -v cpu
```

Также агент выполнит команду в оболочке *shell* и соберет результат, как если бы она была выполнена оператором:

```
$> myScript.pl --h 127.0.0.1 -v cpu
```

Удаленная конфигурация

В версии Enterprise возможно удаленное управление файлами программных агентов из веб-консоли Pandora FMS. Это позволяет централизованно управлять всеми нашими программными агентами без необходимости прямого доступа к системам, в которых они установлены.

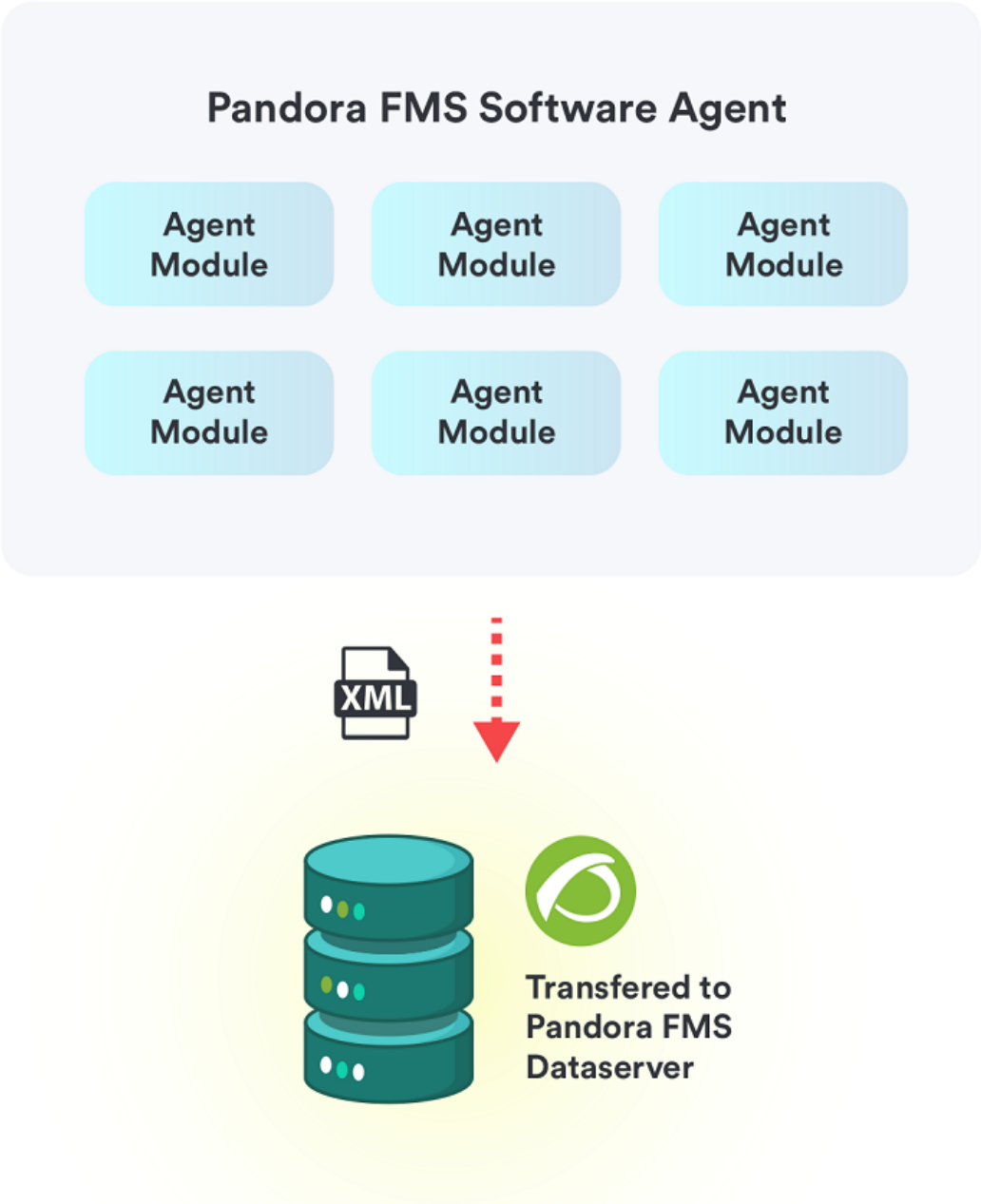
Конфигурация каждого агента хранится на сервере Pandora FMS в двух файлах: <md5>.conf и <md5>.md5, где <md5> - это *хэш* имени программного агента. Эти файлы хранятся, соответственно, в */var/spool/pandora/data_in/conf* и */var/spool/pandora/data_in/md5*. Консоль отвечает за синхронизацию этих файлов на сервере Pandora FMS и соответствующих локаций, где установлен каждый программный агент.

```
 /var/spool/pandora/data_in/conf
```

и

```
/var/spool/pandora/data_in/md5
```

Чтобы включить удаленную конфигурацию, сначала необходимо включить соответствующий параметр в локальном файле конфигурации агента. С этого момента все изменения должны осуществляться с консоли Pandora FMS:

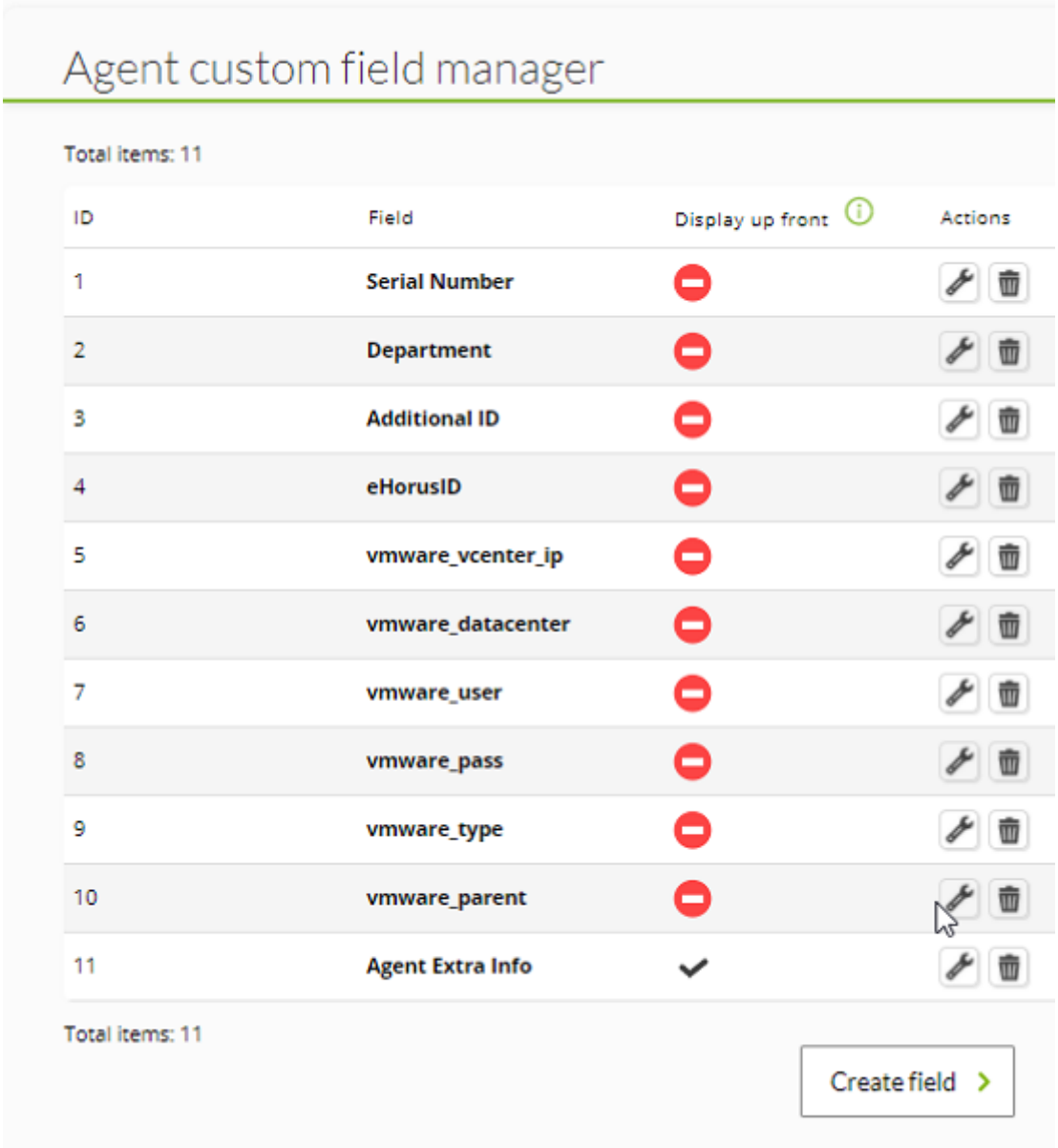


Чтобы включить удаленную конфигурацию, сначала необходимо включить соответствующий параметр в локальном файле конфигурации агента. С этого момента все изменения должны осуществляться с консоли Pandora FMS:

```
remote_config 1
```

После включения удаленной конфигурации агента любые изменения, внесенные локально в файл конфигурации, будут перезаписаны конфигурацией, хранящейся в консоли. Чтобы вернуться к локальному управлению программным агентом, остановите службу ,сбросьте **remote_config** на ноль и снова запустите службу.

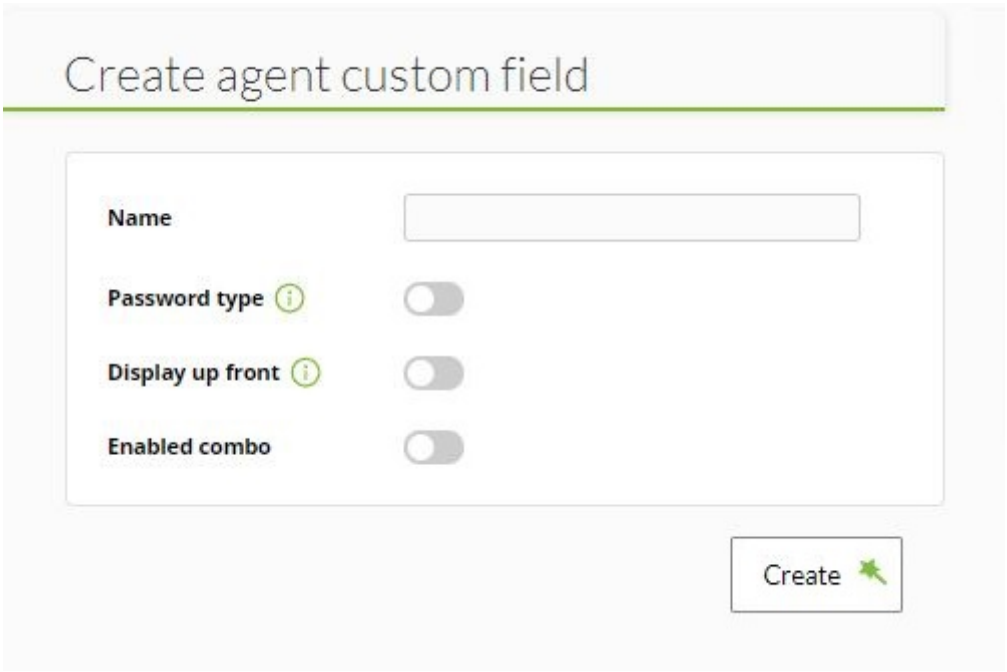
Custom fields



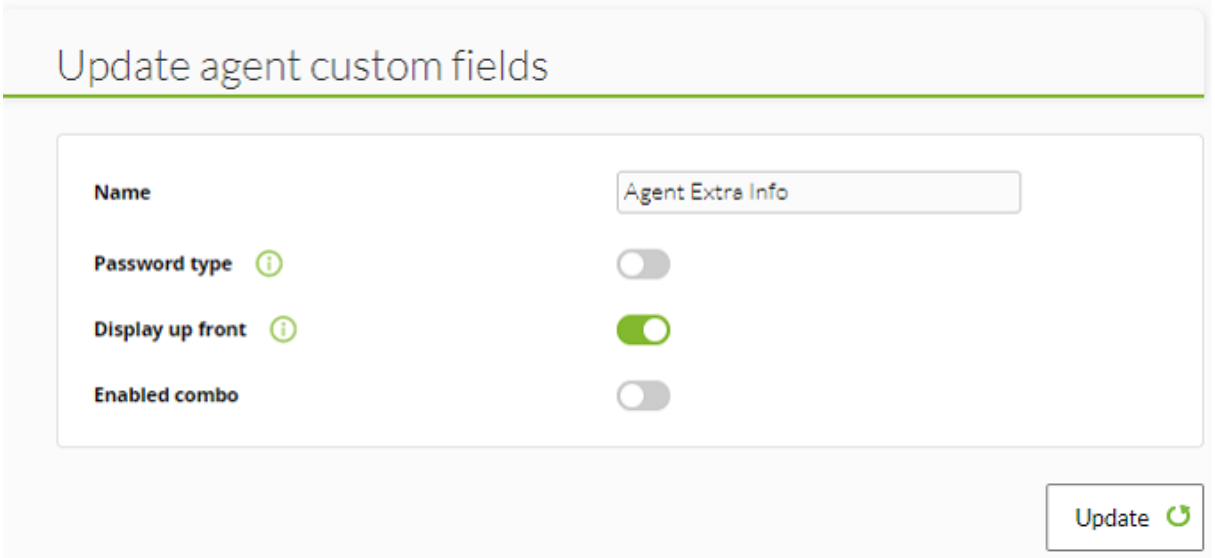
Настраиваемые поля позволяют добавлять агенту дополнительную информацию. Вы можете создать пользовательские поля в **Resources** > **Custom fields**.

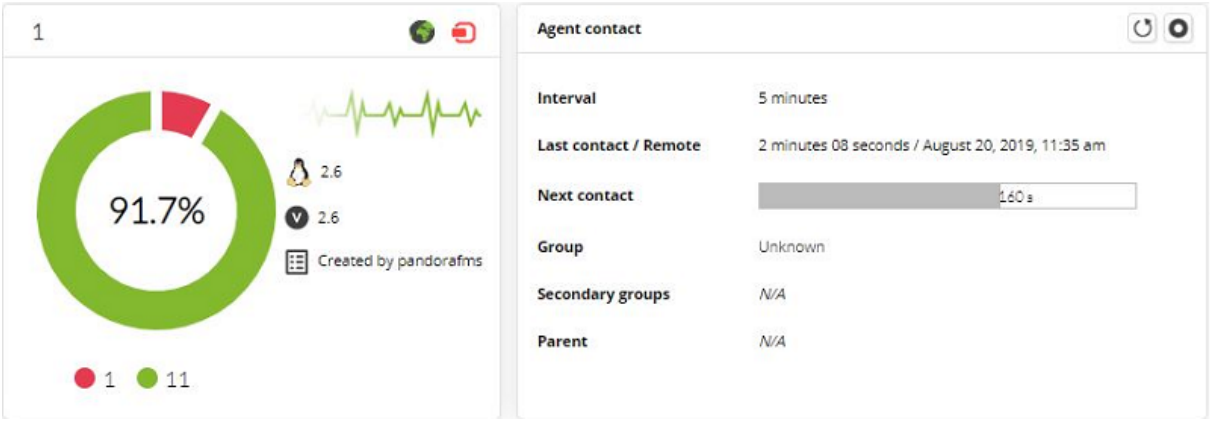
Можно включать ссылки в *custom fields* с помощью тегов [url]ссылка[url] или [url=ссылка]Имя Веб[url].

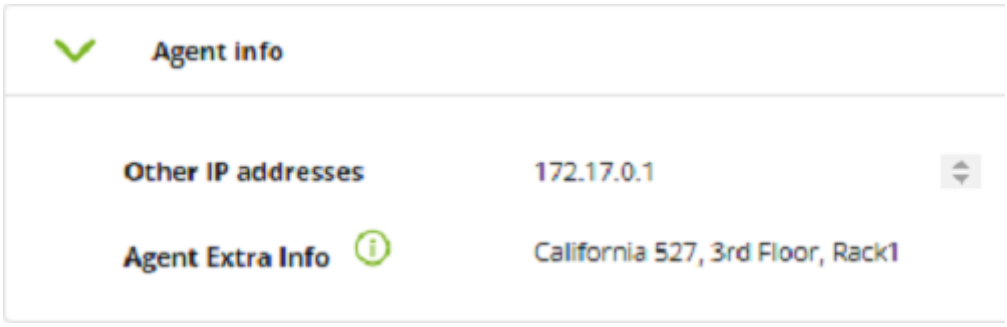
По умолчанию поля **Display up front** и **Enabled combo** отключены:



- При активации поля **Display up front** информация о пользовательском поле будет отображаться в обзоре агента. Кроме того, необходимо активировать этот *токен*, чтобы отправить информацию о *Custom Fields* в Метаконсоль и иметь возможность показать ее при просмотре Метаконсоль и работать через Custom Field View с этими данными:







- **Enabled combo**: После включения в окне конфигурации соответствующего пользовательского поля появится новое поле для ввода значений combo, разделенных запятыми. Этот параметр позволяет активировать настройку выбираемых параметров из выпадающего списка.

Если активирован параметр «Enabled combo», то «Password type» будет отключен.

Поля *custom fields* можно также передать из файла конфигурации агента, используя следующий *token* конфигурации, например:

```
custom_field1_name Model
custom_field1_vale i386
```

Общие параметры конфигурации

Наиболее важные параметры для базовой конфигурации агента (подробнее вПрограммные агенты Pandora FMS):

- **server_ip**: IP-адрес сервера Pandora FMS.
- **server_path**: Путь к папке *incoming* входящих файлов сервера Pandora FMS, по

умолчанию */var/spool/pandora/data_in*.
- **temporal**: Папка, по умолчанию */tmp*.
- **logfile**: файл журнала программного агента, по умолчанию */var/log/pandora/pandora_agent.log*.
- **interval**: интервал выполнения агента, по умолчанию 300 секунд.
- **agent_name**: Имя, по умолчанию *hostname*.
- **debug**: При включении (значение 1) устанавливает режим отладки *debug*, при этом создается копия XML, которая отправляется на сервер и сохраняется во временном каталоге для анализа. Кроме того, в системах MS Windows® по пути установки Агента создается файл `.debug` с подробным *журналом* выполнения каждого модуля. В версиях Enterprise активный режим *debug* отключает удаленную конфигурацию с консоли на Программного Агента.

Активный режим *debug* не предназначен для длительного использования. Это режим для отладки ошибок за короткие промежутки времени. **Важно не забыть отключить его сразу после завершения отладки ошибок.**

Пример в среде ***nix**:

```
server_ip       192.168.1.1
server_path     /var/spool/pandora/data_in
temporal        /tmp
logfile         /var/log/pandora/pandora_agent.log
interval        300
debug           0
agent_name      box01
server_port     41121
transfer_mode   tentacle
remote_config    1
```

Пример в среде **MS Windows®**:

```
server_ip       192.168.1.1
server_path     /var/spool/pandora/data_in
temporal        "C:\Program Files\pandora_agent\temp"
logfile         "C:\Program Files\pandora_agent\pandora_agent.log"
interval        300
debug           0
agent_name      box02
server_port     41121
transfer_mode   tentacle
```

```
remote_config   1
```

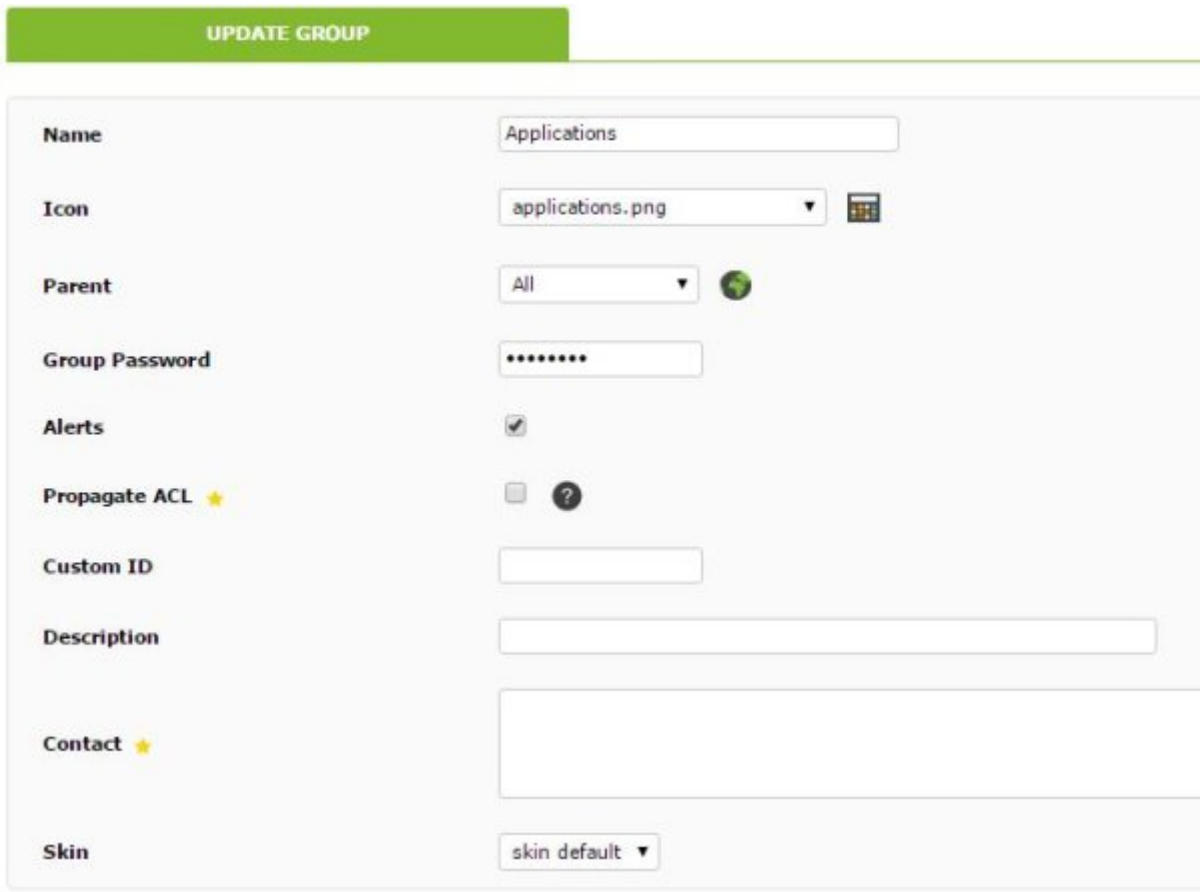
группы, защищенные паролем

По умолчанию, когда агент впервые передает данные на сервер Pandora FMS, он автоматически отправляется в группу, определенную в файле конфигурации агента. Это означает, что в действительности любой может добавить агента в группу, если он знает название группы. Это может стать проблемой, если несколько клиентов совместно используют экземпляр Pandora FMS или если вы хотите контролировать, что находится в каждой группе.

Можно дополнительно настроить пароль для группы через Консоль pandora FMS. Агент не будет добавлен в группу, если в файле конфигурации агента не указан правильный пароль.

Пример

Чтобы настроить пароль для группы, перейдите в *редактор групп* и нажмите *редактировать*, введите пароль группы и сохраните ваши изменения:



Чтобы добавить новый агент в эту группу, отредактируйте файл конфигурации и добавьте следующую опцию конфигурации:

```
group_password <password>
```

Не забудьте перезапустить службу программного Агента, чтобы изменения вступили в силу. Затем агент должен быть правильно создан в Консоли Pandora FMS.

Модули в агентах и Программные Агенты Software Agents

Типы модулей

В зависимости от возвращаемого типа данных:

- **generic_data**: числовые данные и данные с плавающей запятой.
- **generic_data_inc**: тип возрастающих числовых данных. Сохраняет разницу между предыдущими данными и текущими данными, деленными на прошедшее время в секундах, показывая скорость в секунду. Используется для подсчета циклов чего-либо в секунду (Герц), например, записи в журнале/сек, полученные байты/сек, входящие соединения/сек и т. д.
- **generic_data_inc_abs**: тип абсолютных возрастающих числовых данных. Он сохраняет разницу между предыдущими и текущими данными без деления между прошедшими секундами, поэтому значение соответствует общему приращению между двумя выполнениями, независимо от фактора времени. Этот тип данных используется для подсчета количества раз чего-либо, например, записей в журнале, общего количества полученных байтов, количества входящих соединений и т. д.
- **generic_proc**:Тип данных *boolean*, ноль (0) означает False или неправильно (предварительно настроено как «критическое состояние»), а значения больше нуля означают True или правильно.
- **generic_data_string**: алфавитно-цифровой тип данных (текст).
- **async_data**: содержит значение generic_data, но обновляется только при изменении. Асинхронные типы данных не имеют определенной временной периодичности.
- **async_string**: то же самое, что и async_data, но с данными типа generic_string. Полезно для мониторинга поиска в *журналах* или средствах просмотра событий.
- **async_proc**: то же самое, что и async_data, но с типом данных generic_proc (*boolean*).
- **Модуль изображения**: используют в качестве базы модуль строкового типа

(generic_data_string или async_string). Если данные, содержащиеся в модуле, представляют собой изображение в кодировке base64 (заголовок «data:image»), то оно будет идентифицировано как изображение и при просмотре появится ссылка на окно для получения изображения. Кроме того, содержание различных изображений, составляющих сохраненные цепочки, будет отображаться в соответствующей истории.

Интервалы в локальных модулях

Все локальные (или программные агентские) модули имеют в качестве своей «базы» интервал своего агента. Однако они могут принимать значения, превышающие эту базу, если вы измените параметр module_interval на помноженное целое число, больше нуля; например:

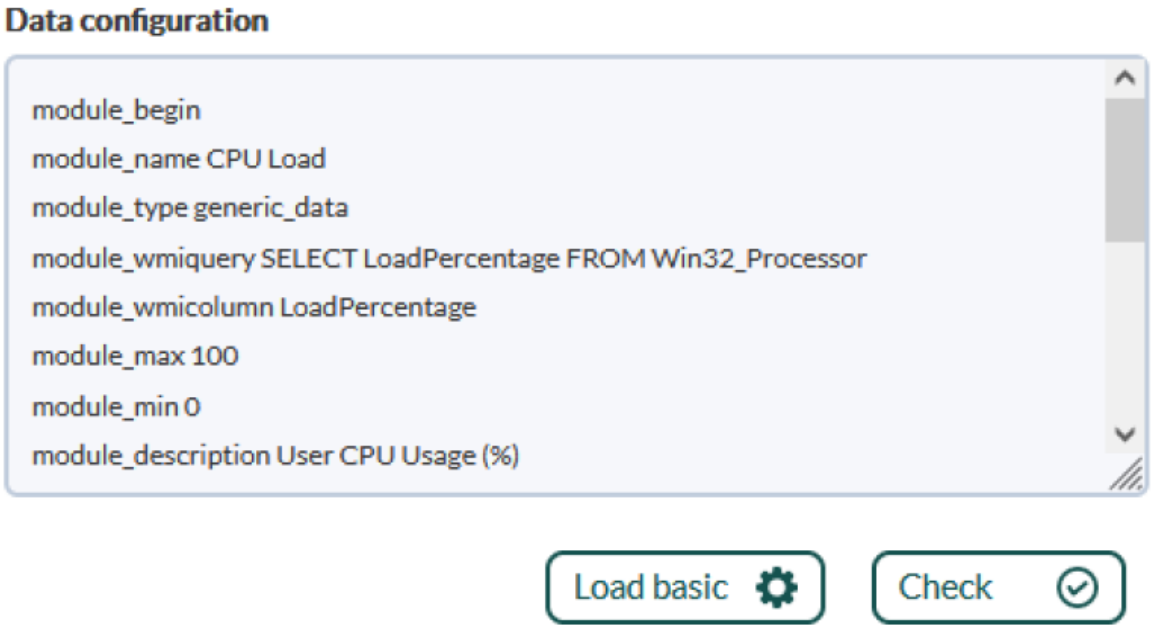
```
module_interval 2
```

Если агент имеет интервал 300, то модуль интервала будет составлять 300×2 (600).

Интерфейс создания модулей

Функция только для версии **Enterprise**; удаленная конфигурация соответствующего программного агента должна быть включена.

Создание локальных модулей в консоли осуществляется через форму, в которой, помимо общей конфигурации любого модуля (пороги, тип, группа и т.д.), имеется текстовое поле, в котором можно указать конфигурационные данные, которые необходимо задать в конфигурационном файле программного агента.



- При нажатии на кнопку **Загрузить базовый (шаблон)**, содержимое *Конфигурация данных* будет удалено с базовым шаблоном, который мы должны изменить в соответствии с необходимостью мониторинга.
- После изменения, нажав **Проверка (синтаксис)**, вы убедитесь, что синтаксис шаблона по-прежнему правильный, однако остальные команды проверяться не будут.

Когда модуль загружаен из локального компонента, он может иметь макросы. Если у него есть макросы, поле конфигурации будет скрыто и появится поле для каждого макроса, см. дополнительную информацию Шаблоны и компоненты

Условный мониторинг

Постусловия

Программный агент поддерживает выполнение команд и *скриптов* в режиме пост-условий. Это означает, что вы можете **осуществлять действия в зависимости от значения, полученного при выполнении модуля**.

Пример 1

Через параметр *module_condition* мы укажем значение (или диапазон значений) и действие, которое нужно выполнить, если полученные данные удовлетворяют условию (использование процессора менее 20%):

```
module_begin
 module_name CPU_Usage_Condition
 module_type generic_data
 module_exec get_cpu_usage.pl
 module_condition <20 add_processes.sh
 module_end
```

Пример 2

Многочисленные условия могут быть заданы для одного и того же модуля, в диапазоне и с минимальным порогом (математически выполняется один или ни один из двух вариантов) :

```
module_begin
 module_name CPU_Usage_Condition
 module_type generic_data
 module_exec get_cpu_usage.pl
 module_condition (90, 100) remove_processes.sh
 module_condition <20 add_processes.sh
 module_end
```

Пример 3

Похоже на предыдущий пример, но вы можете выполнить оба условия или одно, или ни одного (попробуйте с выбранными значениями: если 5, 15 или 30):

```
module_begin
 module_name CPU_Usage_Condition
 module_type generic_data
 module_exec get_cpu_usage.pl
 module_condition <10 start_new_server.sh
 module_condition <20 add_processes.sh
 module_end
```

Предусловия

Параметр **module_precondition** позволяет оценить условие перед выполнением модуля и по результату решить, следует ли выполнять модуль или нет.

Пример 1

В зависимости от использования ЦП, если активных процессов больше десяти, получите процент использования ЦП и сообщите об этом на сервер Pandora FMS:

```
module_begin
```

```
module_name CPU_Usage
module_type generic_data
module_precondition> 10 number_active_processes.sh
module_exec get_cpu_usage.pl
module_end
```

Пример 2

Можно задать несколько предварительных условий для одного и того же модуля, и все они должны быть выполнены:

```
module_begin
 module_name CPU_Usage
 module_type generic_data
 module_precondition> 10 number_active_processes.sh
 module_precondition> 1 important_service_enabled.sh
 module_exec get_cpu_usage.pl
 module_end
```

В этом случае модуль запускается только при наличии более десяти активных процессов и если хотя бы один из них является важным процессом.

Интенсивный мониторинг

Существуют определенные модули, которые имеют особое значение, например, процессы или критически важные в исполнении сервисы. Для более контролируемого наблюдения за этими случаями существует интенсивный мониторинг.

Заключается в предупреждении через более короткий интервал времени о том, что появилась серьезная проблема, при этом нет необходимости сокращать общий интервал агента.

Конфигурация в Программном Агенте:

- **interval**: обязательно, время выборки агента в секундах, является общим интервалом для всех локальных модулей.
- **intensive_interval**: по желанию, время, за которое мы уведомим вас о существовании проблемы.

Конфигурация в модуле:

- **module_intensive_condition = <valor>**: если модуль получит в результате значение <valor>, указанное в этом параметре, он сообщит об этом в интенсивном интервале, определенном выше.

Пример

Служба **sshd** очень важна, поскольку она используется для удаленного подключения с помощью *shell*, нам необходимо контролировать ее работу:

```
intensive_interval 10
 interval 300
```

```
module_begin
 module_name SSH Daemon
 module_type generic_data
 module exec ps aux | grep sshd | grep -v grep | wc -l
 module_intensive_condition = 0
 module_end
```

Если служба отсутствует, она сообщит об этом в ближайшие 10 секунд, если она работает, то она будет сообщать об этом каждые 5 минут (нормальный интервал - 300 секунд).

Запрограммированный мониторинг

Программный агент поддерживает определение запланированных модулей, которые выполняются в определенное время. Используемый синтаксис такой же, как и в файле crontab. Пример определения модуля для запуска каждый понедельник между 12:00 и 15:00 часами:

```
module_begin
 module_name crontab
 module_type generic_data
 module_exec script.sh
 module_crontab * 12-15 * * 1
 module_end
```

Запускать на 10-й минуте каждого часа:

```
module_begin
 module_name crontab
 module_type generic_data
 module_exec script.sh
 module_crontab 10 * * * *
 module_end
```

Если мы используем интервал, из-за которого модуль не сообщает данные, модуль будет переведен в статус «неизвестно». Для таких случаев используйте асинхронные модули.

Удаленные проверки с помощью Программного Агента

Когда главный сервер Pandora FMS не имеет доступа для выполнения удаленных проверок (обычно по причинам безопасности), программный агент способен занять его место для выполнения таких задач и даже может распределить их между *брокерскими агентами*

Проверки ICMP

ICMP или ping-проверки очень полезны для того, чтобы узнать, подключено устройство к сети или нет. Таким образом, один программный агент может контролировать состояние всех устройств простым способом.

Unix

Используя системные команды (все параметры в «командной строке» module_exec):

```
module_begin
module_name Ping
module_type generic_proc
module_exec ping -c 1 192.168.100.54>/dev/null 2>&1; if [ $? -eq 0
]; then echo 1; else echo 0; fi
module_end
```

Примечание: замените «192.168.100.54» на IP-адрес, который будет контролироваться.

MS Windows®.

Параметры должны быть указаны в module_ping_count (количество пакетов, по умолчанию 1) и module_ping_timeout (ограничение по времени в секундах, по умолчанию 1); пример:

```
module_begin
 module_name Ping
 module_type generic_proc
 module_ping 192.168.100.54
 module_ping_count 2
```

```
module_ping_timeout 5
module_end
```

Примечание: module_advanced_options позволяет использовать расширенные опции для ping.exe.

Проверки TCP

Проверки TCP полезны для того, чтобы удостовериться, что порты устройства остаются открытыми, фа также позволяют узнать, подключается ли приложение к сети.

Unix

С помощью команды **nmap** и ее конфигурационных параметров в командной строке, по IP-адресу проверяем, открыт ли порт 80 (время ожидания ответа 5 секунд):

```
module_begin
module_name PortOpen
module_type generic_proc
module_exec nmap 192.168.100.54 -p 80 | grep open> /dev/null 2>&1;
echo $?; if [ $? == 0 ]; then echo 1; else echo 0; fi
module_timeout 5
module_end
```

MS Windows®.

Параметры должны быть указаны в:

- module_tcpcheck: IP-адрес устройства.
- module_port: номер порта.
- module_timeout: время ожидания ответа.

Пример:

```
module_begin
 module_name PortOpen
 module_type generic_proc
 module_tcpcheck 192.168.100.54
 module_port 80
 module_timeout 5
 module_end
```

Проверки SNMP

Проверки SNMP очень распространены в мониторинге сетевых устройств для проверки состояния интерфейсов, байтов ввода/вывода и так далее.

Unix

Например, с помощью команды **snmpget**:

```
module_begin
 module_name SNMP get
 module_type generic_data
 module_exec snmpget 192.168.100.54 -v 1 -c public
.1.3.6.1.2.1.2.2.1.1.148 | awk '{print $4}'
 module_end
```

Этот модуль возвращает значение OID .1.3.6.1.2.1.2.2.1.1.148 *хоста* 192.168.100.54.

MS Windows®.

Настройки параметров:

- module_snmpversion [1,2c,3]: Версия SNMP (по умолчанию 1).
- module_snmp_community <community>: Сообщество SNMP (по умолчанию публичное).
- module_snmp_agent <host>: Агент SNMP цель.
- module_snmp_oid <oid>: OID цель.
- module_advanced_options: Дополнительные параметры для snmpget.exe.

Пример, который делает то же самое, что и предыдущий пример:

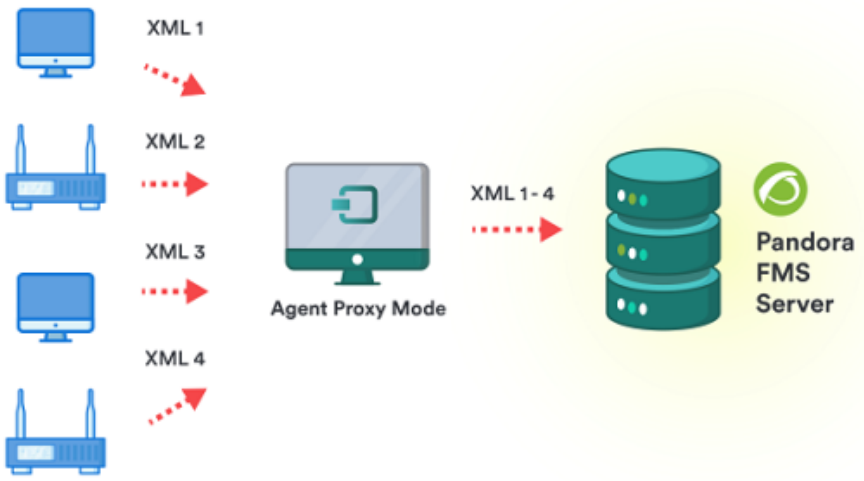
```
module_begin
 module_name SNMP get
 module_type generic_data
 module_snmpget
 module_snmpversion 1
 module_snmp_community public
 module_snmp_agent 192.168.100.54
 module_snmp_oid .1.3.6.1.2.1.2.2.1.1.148
 module_end
```

Режим Прокси

Для использования прокси-режима агента Pandora

FMS в Linux/Unix **не может выполняться пользователем root**, поэтому необходима специальная установка агента Pandora FMS. Для этого, пожалуйста, обратитесь к
Пользовательская установка агента

Программные агенты Pandora FMS имеют Режим Прокси, который позволяет им перенаправлять файлы данных, созданные другими программными агентами, на сервер Pandora FMS. Программный агент, работающий в режиме прокси, также может выполнять задачи мониторинга.



Режим прокси был создан для локальных сетей, в которых только один компьютер выходит в Интернет, где расположен сервер Pandora FMS. Необходимо проводить мониторинг с помощью Программных Агентов остальные компьютеры в этой сети; остальные компьютеры будут связываться с прокси, а не с сервером. Режим прокси также поддерживает функции *Удаленная конфигурация* и *Коллекции файлов*.

Настройки параметров:

- **server_ip**: IP-адрес сервера Pandora FMS.
- **proxy_mode**: активирован (1) или деактивирован (0).
- **proxy_max_connection**: количество одновременных прокси-соединений, по умолчанию 10.
- **proxy_timeout**: время ожидания ответа прокси-сервера, по умолчанию 1 секунда.
- **proxy_address**: адрес, по которому прослушивается прокси-сервер.

- **proxy_port**: порт, на котором прослушивается прокси-сервер.

Пример:

```
server_ip 172.17.100.230
 proxy_mode 1
 proxy_max_connection 20
 proxy_timeout 3
```

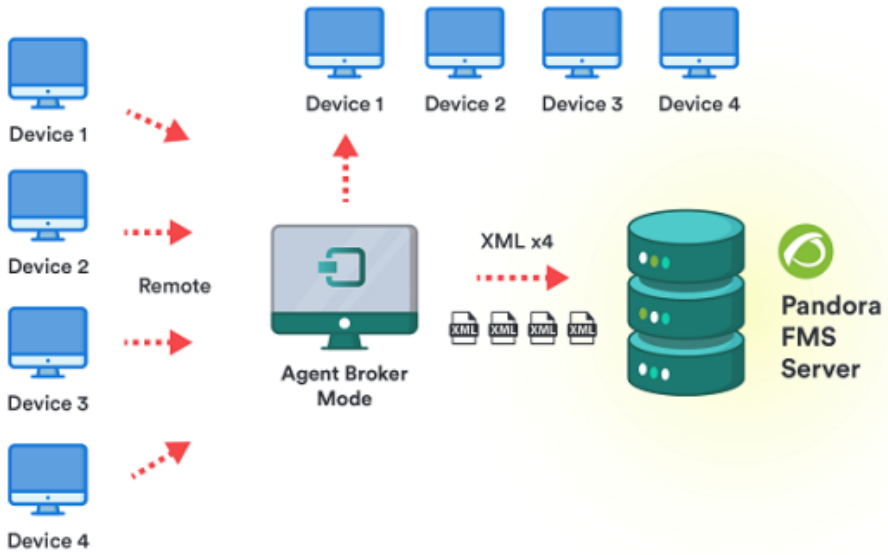
Для перенаправления соединения программного агента достаточно установить в качестве адреса сервера Pandora FMS адрес прокси-агента с активированным режимом прокси.

Например, программный агент в режиме Proxy имеет IP-адрес 192.168.100.24, остальные программные агенты должны быть сконфигурированы:

```
server_ip 192.168.100.24
```

Режим Broker

Режим брокера программных агентов позволяет одному агенту выполнять проверки и управлять конфигурацией, как если бы это было несколько разных агентов.



Когда Режим Брокера включен в Программном Агенте, создается новый файл конфигурации. С этого момента оригинальный программный Агент и новый Брокер

будут управляться отдельно со своими независимыми конфигурационными файлами, как если бы это были два совершенно отдельных программных агента на одном устройстве.

Основные цели:

- Отправка локальных данных в качестве другого агента. Очень полезно для мониторинга экземпляров программного обеспечения как независимых агентов.
- Отправляйте данные, собранные в ходе удаленных проверок, на другие устройства, как если бы на них был установлен программный агент.

Для создания брокера достаточно добавить строку с парой **broker_agent <nombre_broker>**. Вы можете создать столько агентов Брокера, сколько захотите, добавив соответствующие строки. *broker_agent*:

```
broker_agent dev_1
broker_agent dev_2
```

После создания брокеров будут созданы их конфигурационные файлы *dev_1.conf* и *dev_2.conf* с тем же содержанием, что и оригинальный программный агент, но с соответствующим именем. Добавляя или удаляя модули из конфигурационных файлов *dev_1.conf* и *dev_2.conf*, мы можем настроить проверки, выполняемые брокерами.

В консоли Pandora FMS **брокеры воспринимаются и управляются как независимые агенты**, поэтому если у нас установлен программный агент с двумя брокерами, то в консоли мы увидим три разных агента с их модулями, конфигурациями и т.д.

ПРИМЕЧАНИЕ: экземпляры режима брокера не могут использовать коллекции. Если вы хотите использовать коллекции, вы должны распространять их и/или использовать в агенте «real» который используется как база в агенте брокера, а не в одном из его экземпляров.

Модули, сохраняющие данные в памяти между выполнениями (module_logevent и module_regexp в MS Windows®), не работают, когда настроены агенты брокера.

Примеры использования режима брокера

Мониторинг локальной базы данных в качестве другого агента

Например, установлен программный агент, который выполняет мониторинг процессора, памяти и диска компьютера, на котором также запущена база данных. Для независимого мониторинга мы добавляем строку:

```
broker_agent DBApp
```

Таким образом мы создаем агента брокера с именем DBApp, который генерирует конфигурационный файл dbapp.conf. Там мы добавляем, для мониторинга базы данных (количество подключенных пользователей и количество медленных запросов):

```
module_begin
module_name Num Users
module_type generic_data
module_exec get_db_users.pl
module_end

module_begin
module_name Num slows queries
module_type generic_data
module_exec get_db_slows_queries.pl
module_end
```

Консоль Pandora FMS покажет один с именем устройства и модулями CPU, памяти и дисков и, кроме того, другой под названием DBApp с модулями Num Users и Num slows queries.

Удаленный мониторинг устройств с помощью брокеров

Например, на устройстве с MS Windows® установлен программный агент, который следит за процессором, памятью и диском. Нам нужно провести мониторинг *router* с IP 192.168.100.54 без установки агента внутри. Для этого мы создаем брокера с параметром:

```
broker_agent routerFloor5
```

Для этого мы создаем агента брокера с именем routerFloor5. Затем в файле routerFloor5.conf измените строки для размещения доступных модулей ping и snmp:

```
module_begin
 module_name Ping
 module_type generic_proc
 module_ping 192.168.100.54
 module_ping_count 2
```

```
module_ping_timeout 500
module_end

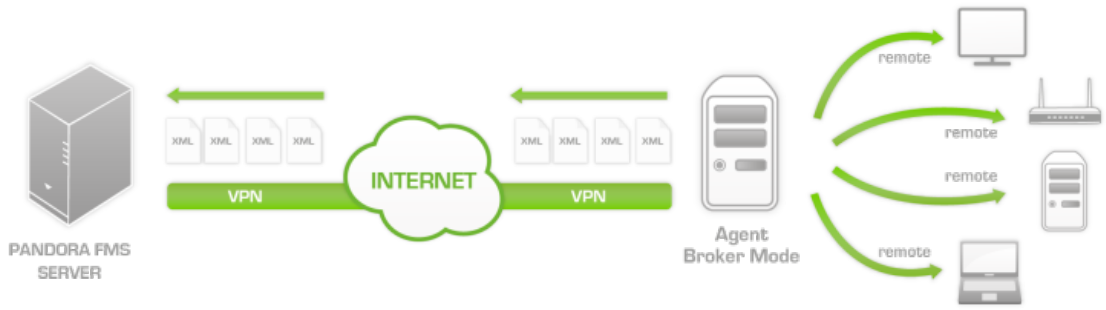
module_begin
module_name Eth 1 up
module_type generic_data
module_snmpget
module_snmpversion 1
module_snmp_community public
module_snmp_agent 192.168.100.54
module_snmp_oid .1.3.6.1.2.1.2.2.1.1.1
module_end

module_begin
module_name Eth 2 up
module_type generic_data
module_snmpget
module_snmpversion 1
module_snmp_community public
module_snmp_agent 192.168.100.54
module_snmp_oid .1.3.6.1.2.1.2.2.1.1.2
module_end
```

В веб-консоли будут показаны два агента: один - устройство Windows с модулями CPU, памяти и диска, а другой - *routerFloor5* с модулями: «Ping», «Eth 1 up» у «Eth 2 up».

Удаленный мониторинг недоступных сетей

В некоторых ситуациях необходимо удаленно контролировать устройства, но Удаленный Сервер Pandora FMS не может получить прямой доступ к элементам.



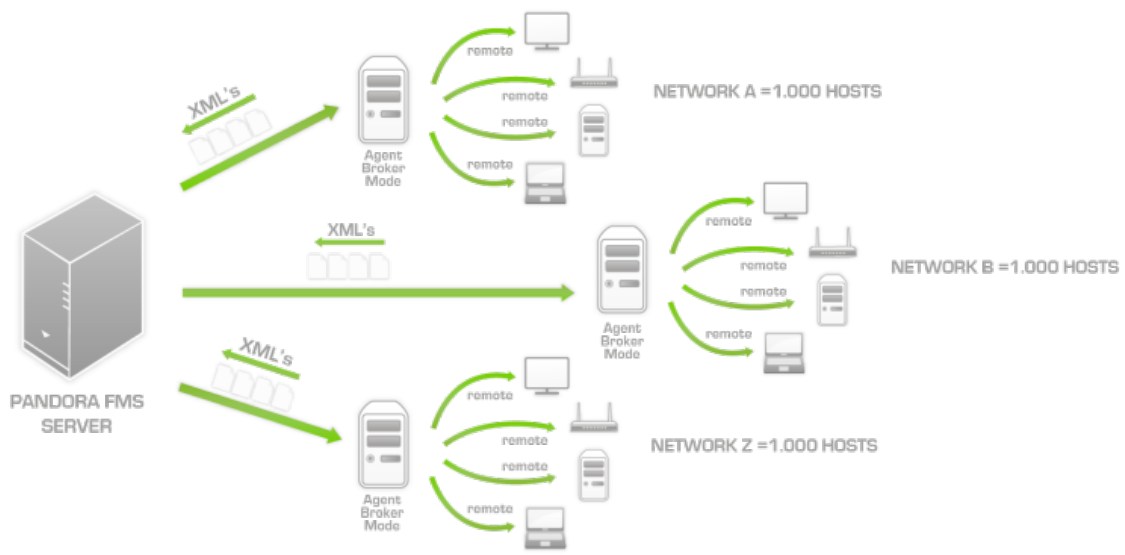
Программный агент в режиме брокера позволяет отправлять XML-файлы на сервер

Pandora, как если бы это было несколько разных устройств. Для этого мы добавим столько брокеров, сколько устройств нужно контролировать, например:

```
broker_agent device_1
broker_agent device_2
broker_agent device_3
broker_agent device_4
...
```

После создания брокеров мы можем настроить мониторинг для каждого устройства, обратившись к конфигурационным файлам каждого брокера, как уже объяснялось в описании каждого программного агента в режиме удаленной проверки.

Распределение нагрузки мониторинга с помощью брокеров



Вместимость удаленного сервера Pandora FMS составляет около 2000 агентов, работая с агентами-брокерами, мы можем увеличить ее до 3000 и освободить основной сервер от большей части работы. На графике каждая из сетей имеет Программного Агента с активированным режимом Broker, в котором мы создадим столько брокеров, мониторинг скольких устройств нам нужно провести. Например, конфигурация для агента *Broker_Agent_Net_A* будет следующей:

```
broker_agent device_1
 broker_agent device_2
 broker_agent device_3
 broker_agent device_4
 ...
```

Кроме того, для каждого из брокеров мы добавляем соответствующие модули для мониторинга устройств, как объяснялось выше.

Инвентаризация с помощью программного агента Software Agent

Программный агент Pandora FMS поддерживает функции инвентаризации как оборудования, так и программного обеспечения. Система инвентаризации позволяет вести богатую историю процессоров, карт, оперативной памяти, патчей, программного обеспечения и т.д., используемых в серверах компании. Также можно генерировать предупреждения в случае изменений в инвентаризации, например, несанкционированной замены жесткого диска или деинсталляции приложения.

Для получения дополнительной информации посетите раздел Локальная инвентаризация с помощью программных агентов.

Удаленные действия через UDP

Программный агент способен принимать удаленные запросы и выполнять команды.

Помните, что UDP по своей природе небезопасен (но эффективен для отправки сообщений без ущерба для определенного ответа).

Чтобы сервер PFMS мог отправлять команды находящимся под его опекой Программным агентам, нужно настроить:

- **udp_server**: ноль по умолчанию, установите значение 1 (один), чтобы включить эту функцию.
- **udp_server_port**: порт прослушивания в программном Агенте.
- **udp_server_auth_address**: IP-адрес сервера Pandora FMS

Перезапустите Software Agent, чтобы изменения вступили в силу.

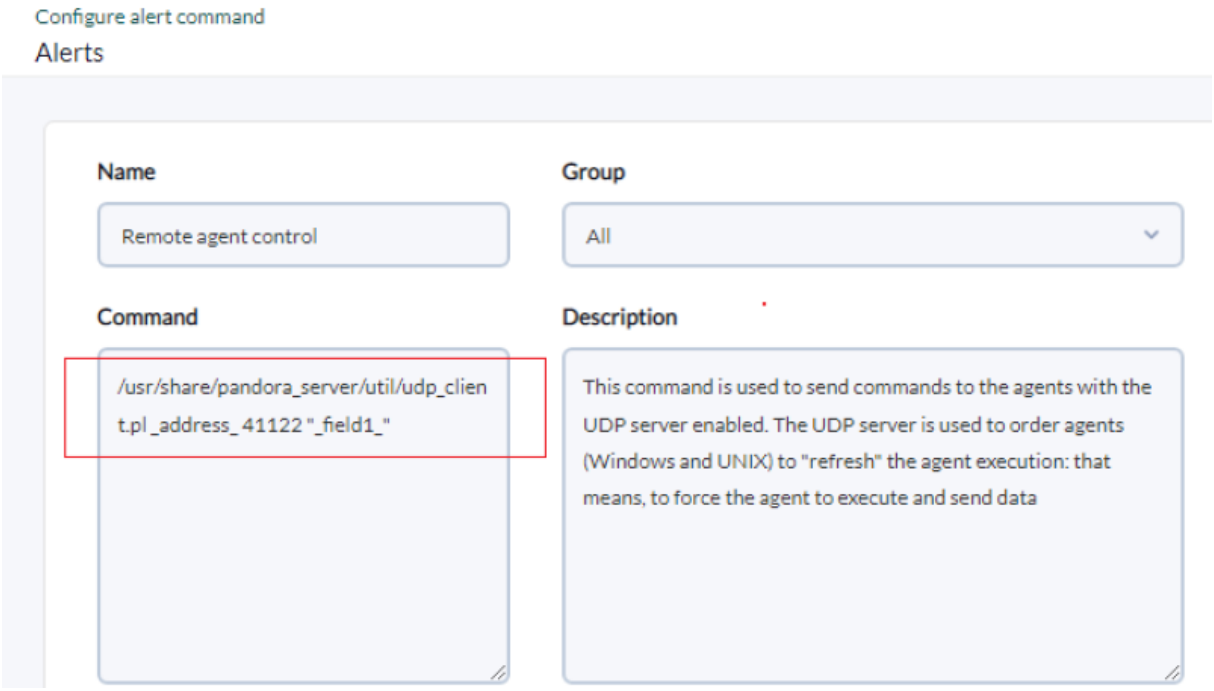
Хотя его можно установить на *0.0.0.0.0* для приема из всех источников, такая практика не рекомендуется. Если у вас несколько серверов PFMS и/или вы используете IPv6, вы можете разместить различные IP-адреса, разделенные запятыми. Например, если вы имеете в IPv6:`2001:0db8:0000:130F:0000:0000:087C:14`

0B и его сокращение
2001:0db8:0:130F::87C:140B, используйте оба
варианта, разделенные запятыми.

Как запросить перезапуск службы у программных агентов Software Agents

Вам следует использовать *скрипт* **udp_client.pl**, который присутствует в сервере Pandora FMS и обычно находится в */usr/share/pandora_server/util*. Вы можете выполнить его из командной строки или использовать его в предупреждении, используя команду из предустановленную на консоли **«Remote agent control»**.

В этом скрипте также есть стандартное действие предупреждения с именем *Restart agent*, использующее действие *REFRESH AGENT*.



Затем можно принудительно запустить предупреждение или запустить неправильное состояние модуля для того, чтобы предупреждение запустилось, и, таким образом проверить конфигурацию.

Пользовательские удаленные действия

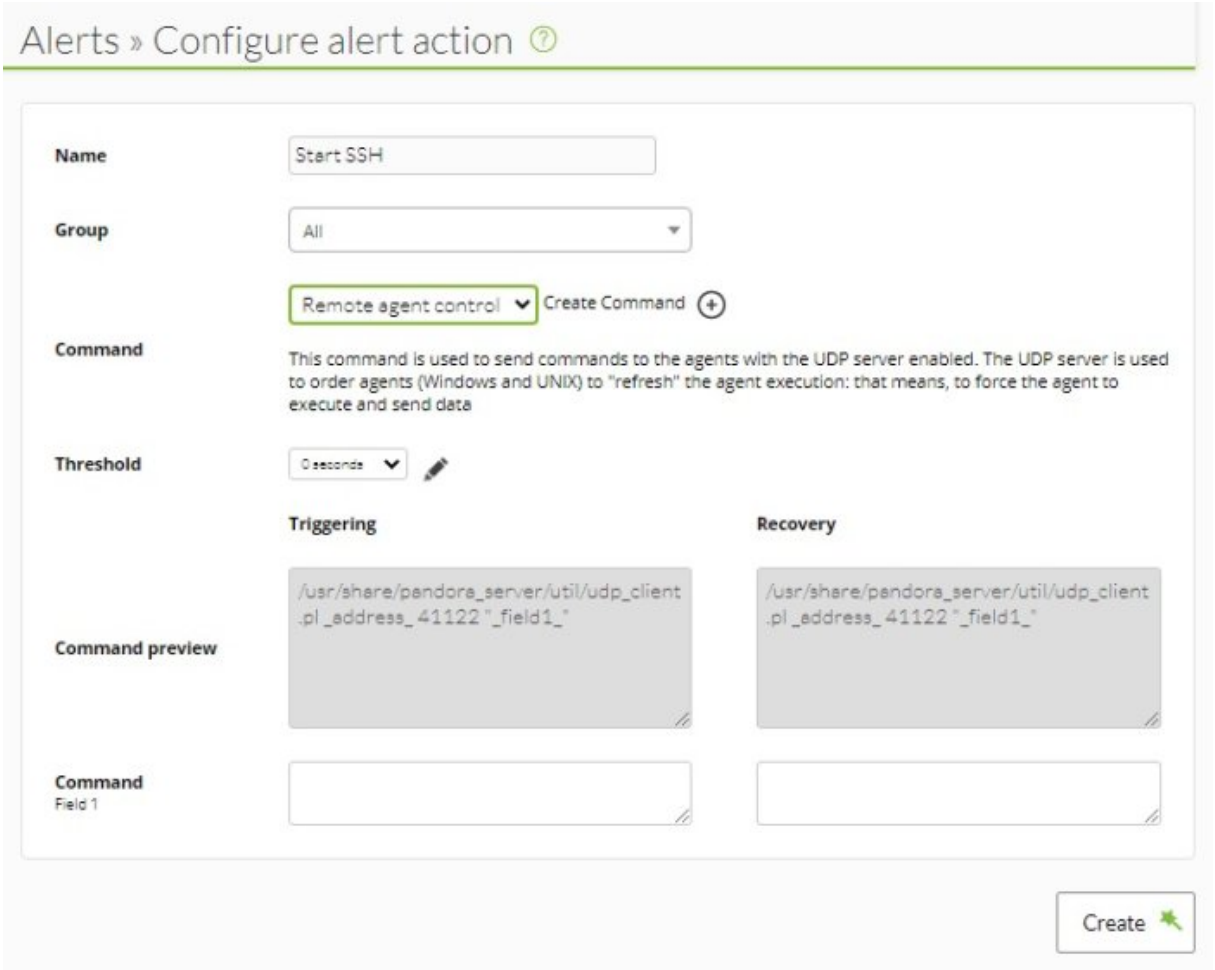
В дополнение к действию перезапуска службы Программного Агента можно указать пользовательские действия. Для этого необходимо добавить строку для каждой выполняемой команды по следующей схеме:

```
process_<nombredelaorden>_start comando
```

Например, если нам нужна удаленная команда для запуска службы sshd:

```
process_sshd_start /etc/init.d/sshd start
```

Следующим шагом необходимо создать действие предупреждения в Pandora FMS Console для каждой созданной вами удаленной команды. Командой для использования будет «Remote agent control» (созданная по умолчанию, она подготовлена для отправки UDP команд) и вставьте в поле 1 «START PROCESS sshd»:



Затем создайте новое ручное оповещение с новым действием в агенте, чью службу **sshd** вы хотите запустить; при принудительном оповещении будет отправлен приказ, и программный агент запустит службу.

Вы также можете создавать команды, вызывающие *скрипты*. Это позволяет выполнять большое количество удаленных действий над

программным агентом одним нажатием кнопки.

Плагины в Программных Агентах Software Agents

Они характеризуются выполнением сложных расширенных проверок со стороны программных агентов, **могут возвращать в качестве результата несколько модулей** вместо одного значения. В отличие от серверных *плагинов*, которые выполняются сервером Pandora FMS, программные агенты возвращают свои данные в XML, сообщая об одном или нескольких модулях одновременно.

Выполнение в системах Windows

В MS Windows® все *плагины* по умолчанию запрограммированы на **VBScript**, для их запуска необходимо использовать соответствующий интерпретатор с указанием полного пути.

Примеры:

```
 module_plugin cscript.exe //B
"%ProgramFiles%\Pandora_Agent\util\logevent_log4x.vbs" Aplicacion
System 300
 module_plugin cscript.exe //B
"%ProgramFiles%\Pandora_Agent\util\df.vbs"
 module_plugin cscript.exe //B
"%ProgramFiles%\Pandora_Agent\util\ps.vbs" iexplore.exe myapp.exe
```

Программный Агент для Windows поставляется с несколькими *плагинами*, готовыми к использованию.

Выполнение в системах Unix

Unix *плагины* по умолчанию ищутся в каталоге агента, в /etc/pandora/plugins, поэтому их вызывают, а затем передают необходимые параметры:

```
  module_plugin grep_log /var/log/syslog Syslog .
  module_plugin pandora_df tmpfs /dev/sda1
```

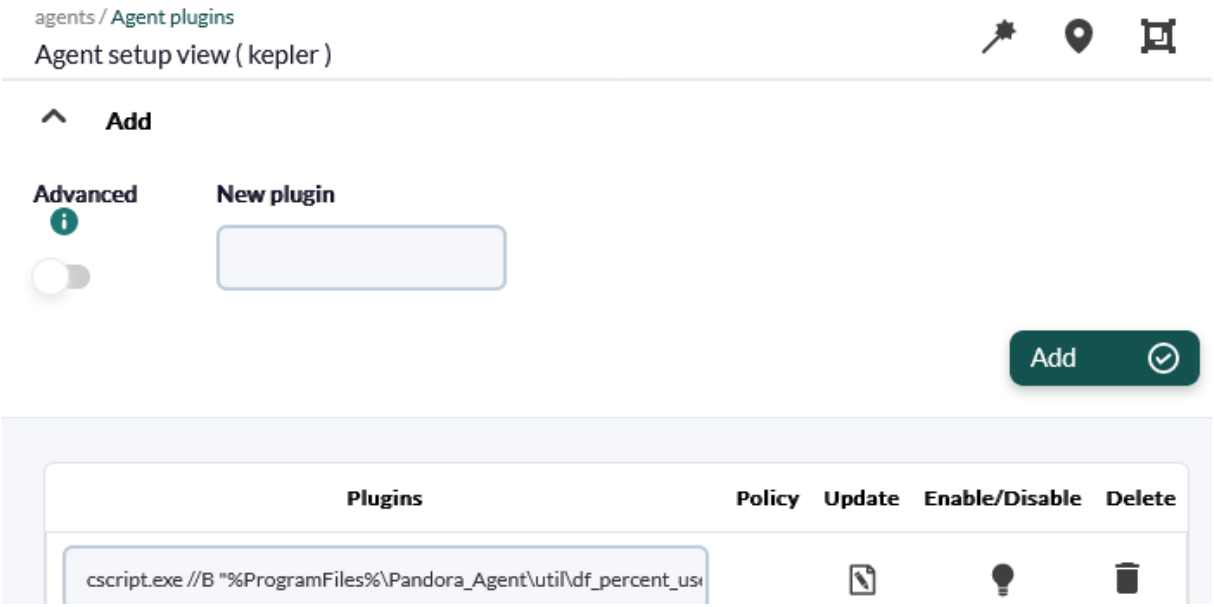
Программный агент Unix поставляется с несколькими готовыми к запуску плагинами.

Управление плагинами Программного Агента из Консоли

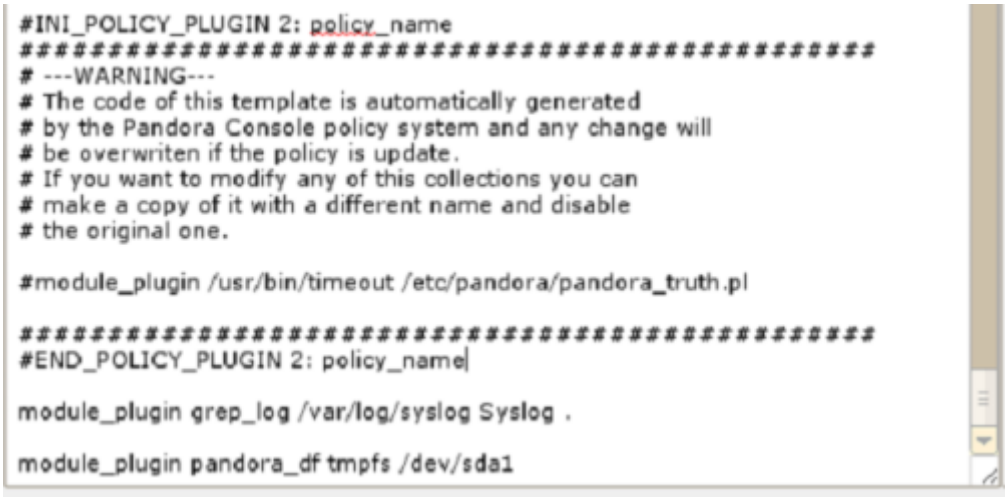
В версии **Enterprise** возможно управление без редактирования непосредственно файла конфигурации. При включении удаленной конфигурации программный агент при его административном просмотре будет иметь вкладку редактора плагинов.



Этот раздел показывает список *плагинов*, активных в агенте, и позволяет удалять, добавлять и отключать их. В случае *плагинов* политики, может быть полезно отключить их, поскольку при повторном применении этой политики, они останутся отключенными.



плагины, управляемые в этом редакторе, можно также редактировать из файла конфигурации агента.



Пример 1

Плагин*для программного агента* может возвращать часть данных или их группу. Примером *плагина*, возвращающего данные, является *ps.vbs* в среде Windows, который проверяет, запущен ли процесс.

```
module_plugin cscript.exe //B
"%ProgramFiles%\Pandora_Agent\util\ps.vbs" IEXPLORE.EXE
```

Результатом будет модуль, который возвращает 0, если процесс не активен, и 1, если он активен:

```
<module>
     <name><![CDATA[IEXPLORE.EXE]]></name>
     <description><![CDATA[Process IEXPLORE.EXE
status]]></description>
     <data><![CDATA[1]]></data>
 </module>
```

Пример 2

Плагин *df.vbs* в среде Windows возвращает свободное пространство на каждом устройстве хранения с помощью следующей команды:

```
module_plugin cscript.exe //B
"%ProgramFiles%\Pandora_Agent\util\df.vbs"
```

Результат:

```
<module>
```

```
     <name><![CDATA[C:]]></name>
     <description><![CDATA[Drive C: free space in
MB]]></description>
     <data><![CDATA[805000]]></data>
 </module>

 <module>
     <name><![CDATA[D:]]></name>
     <description><![CDATA[Drive D: free space in
MB]]></description>
     <data><![CDATA[90000]]></data>
 </module>
```

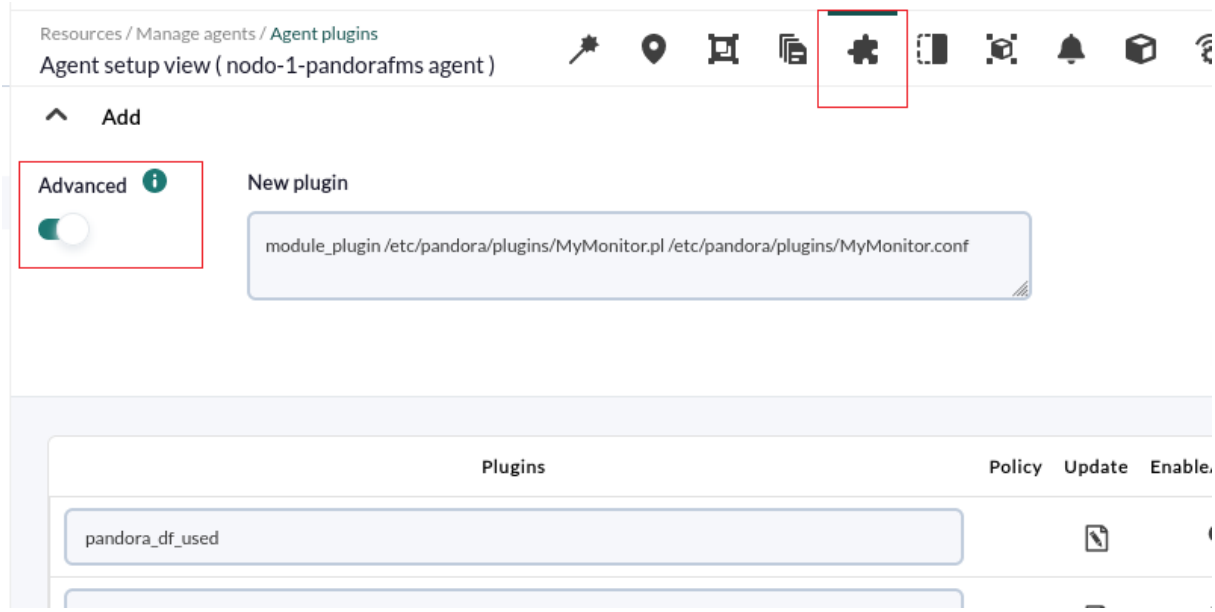
Управление расширенными плагинами Software Agent из Консоли

версия NG 750 или выше.

Можно добавить *токен* в конфигурацию агента *плагинов*, который при включении позволяет 'инкапсулировать' определения *plugins* внутри тегов `module_begin` и `module_end`.

Этот *токен* позволяет вам вставлять блоки конфигурации, такие как `module_interval` или `module_crontab`, среди прочих.

Чтобы включить этот *токен*, просто прокрутите страницу в управлении агентами до плагинов агентов, и в верхней части конфигурации вы найдете его под названием *Advanced*.



Как создавать пользовательские плагины для Agent Software

Плагины могут быть созданы на любом языке программирования. Следует принимать во внимание только общие нормы и специальные нормы для его разработки.

Использование плагинов Nagios из программного агента

Nagios имеет большое количество *плагинов*, которые вы можете использовать с Pandora FMS. Одним из способов это сделать является использование удаленных *плагинов* с Plugin Server, применяя совместимость с Nagios. Но таким образом вы получите только их состояния, потому что он не использует описательный вывод, который имеют некоторые *плагины* для Nagios.

Для этого была создана обертка *wrapper* плагинов Nagios, которая справляется с этой ситуацией. *Обертка* поставляется с агентом Unix 3.2 по умолчанию, а в среде Windows необходимо загрузить ее эквивалент из Библиотека ресурсов Pandora FMS.

Общая эксплуатация

Обертка *wrapper* выполняет плагин Nagios, используя его исходные параметры и преобразуя его вывод в полезные данные для Pandora FMS. Она содержит два типа информации:

- Информация о статусе с учетом уровней ошибок Nagios: NORMAL (1), CRITICAL (0), WARNING (2), UNKNOWN () и другие (4). По умолчанию он будет использовать proc модуль, поэтому значения NORMAL (нормальный) и CRITICAL (критический) работают «по умолчанию»; если вы хотите иметь информацию о WARNING (внимание проблема) и других значениях, вы должны установить пороговые значения модуля вручную.
- Информация описательного типа: Как правило, строковая информация. Она будет помещена в поле описания модуля, например:

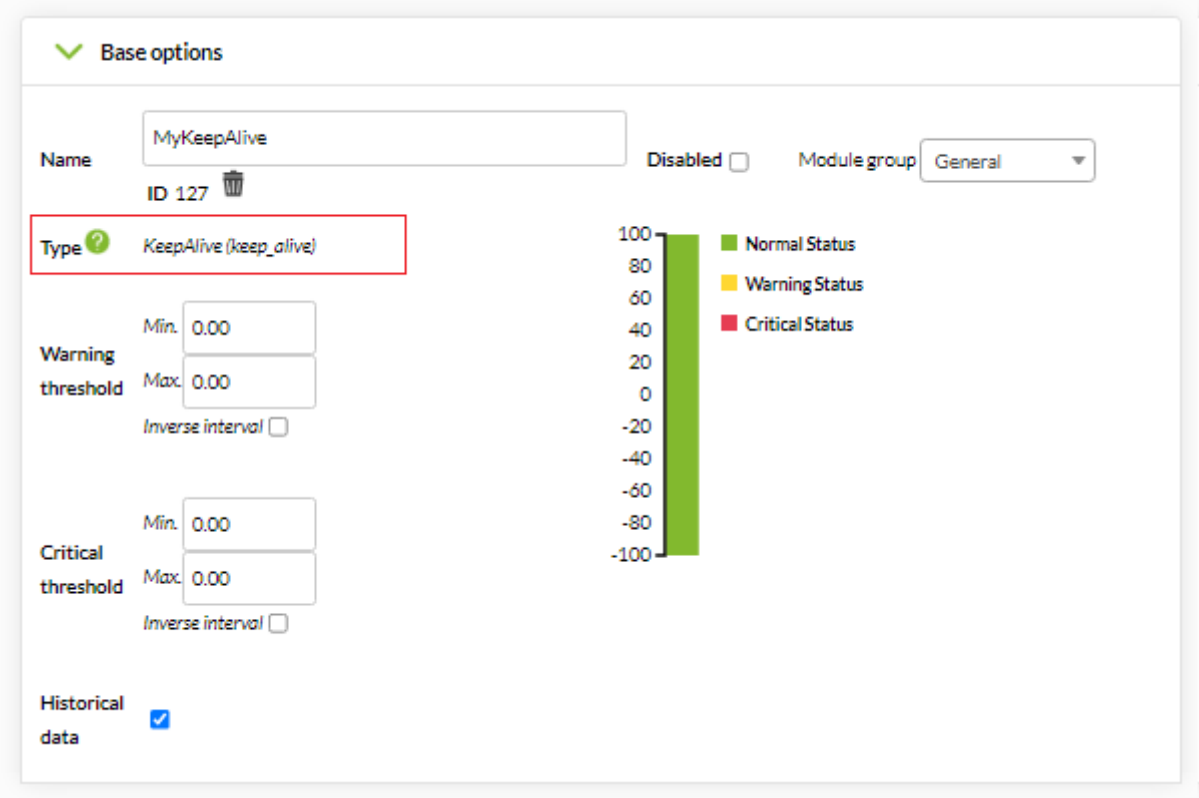
```
<![CDATA["OK: successfully logged in"]]>
```

Мониторинг с помощью KeepAlive

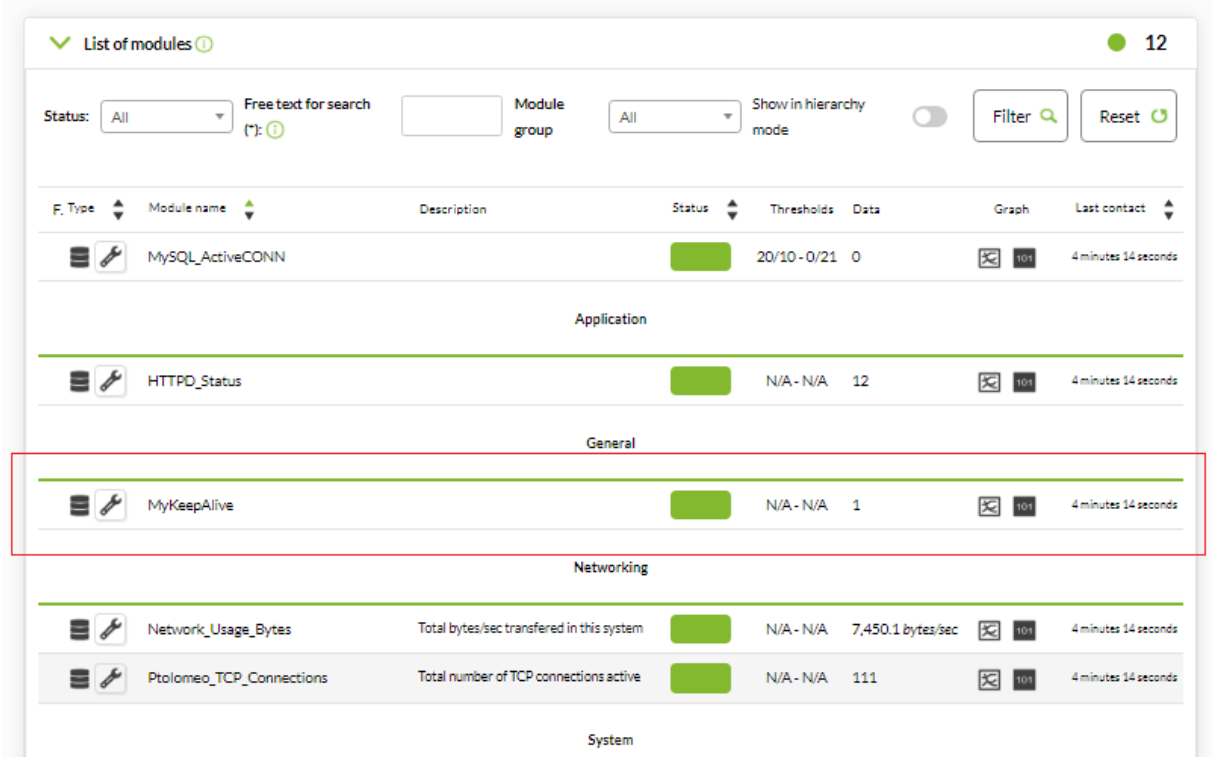
Особенным модулем в Pandora FMS является тип под названием `keep_alive`, используемый для оповещения, если программный агент перестал отправлять информацию (см. выше Удаленные действия через UDP). Это предупреждение возникает, если вы не обновили дату последнего контакта в течение удвоенного интервала времени; она срабатывает и переводит монитор в Критический режим.

Модуль KeepAlive может быть создан только из консоли (даже если у нас не включена удаленная конфигурация), и он не оставляет никаких следов в файле `pandora_agent.conf`

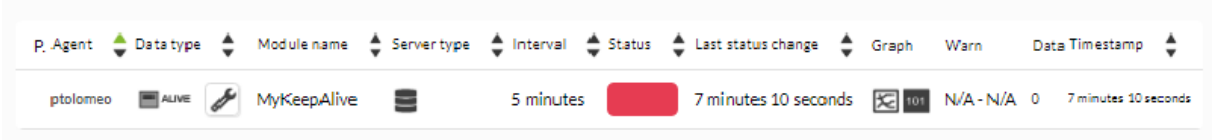
Создание нового модуля типа «KeepAlive»:



Работа в статусе «NORMAL» (зеленый), «NOT INITIALIZED» (синий):

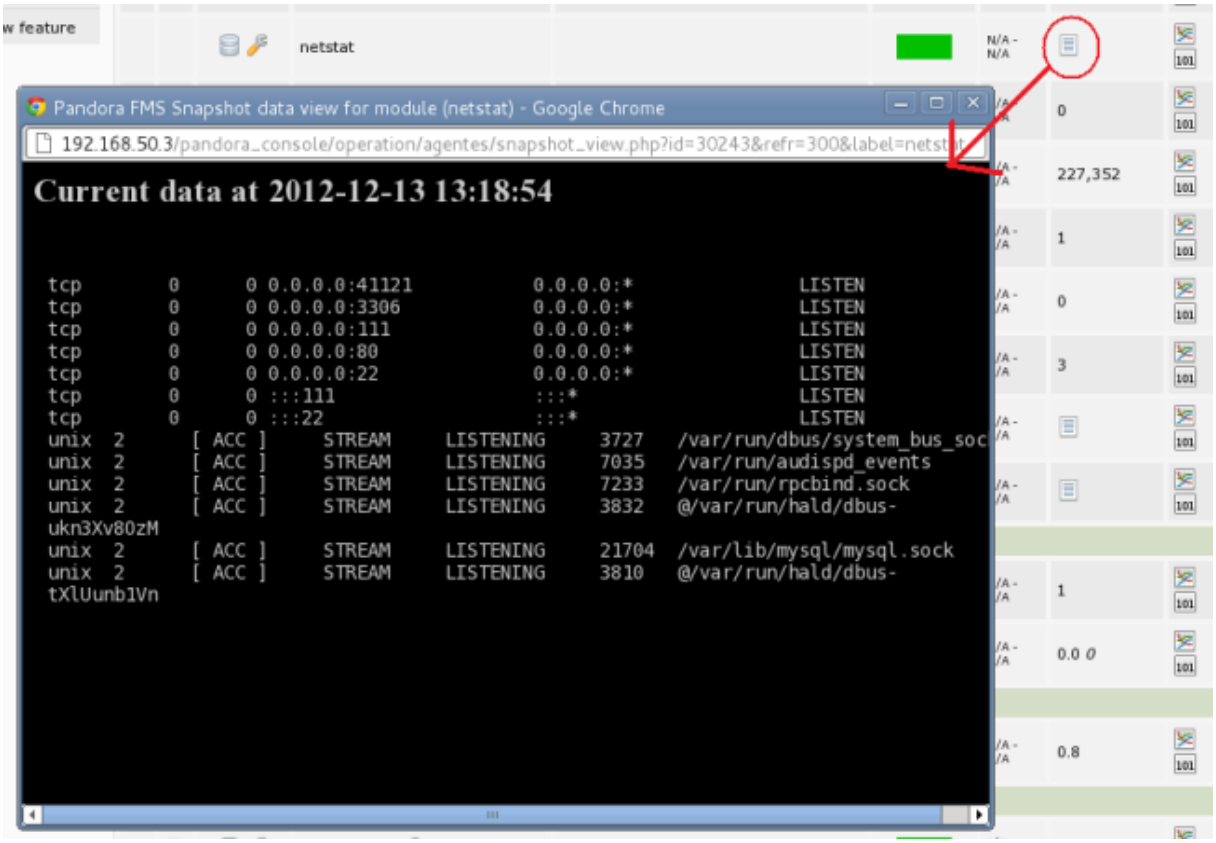


Если агент перестает отправлять данные (в данном примере интервал составлял 1 минуту), то он автоматически переходит в статус CRITICAL (красный). Чтобы просмотреть модули этого типа, можно перейти в левое боковое меню и выбрать пункт **Monitoring → Views → Monitor detail** и в диалоговом окне фильтра поле **Data type** seleccione **KeepAlive** и затем нажмите на **Show**:



KeepAlive он ведет себя как любой другой модуль: с ним можно связать оповещение и использовать его для любых других элементов, таких как отчеты, карты и т.д..

Мониторинг командных снимков (командные снимки)



Comandos que presenten salidas extensas, como *top* o *netstat -n* pueden ser capturados completamente por un módulo y reproducidos tal cual. El módulo debe configurarse como tipo texto.

Чтобы это работало таким образом, необходимо правильно настроить как Консоль Pandora (*setup*), так и агента, который собирает эту информацию, убедившись, что это *необрабатываемый текст*.

В консоли должна быть активирована опция:



Мониторинг и визуализация изображений

Этот метод позволяет определять строковые модули (`generic_data_string` или `async_string`), которые содержат изображения в текстовом формате с кодированием base64 и может вывести это изображение вместо конкретного результата. Это хранится в виде текстовой информации, а отображается по-другому, не как простые данные, а путем реконструкции изображения, когда вы нажимаете на специальный значок для захвата изображения:



Для захвата этих изображений достаточно написать плагин, передающий все данные, сгенерировать необходимые XML-теги и запустить плагин как таковой с помощью директивы *module_plugin*. Пример:

```
#!/bin/bash
echo "<module>"
echo "<name>Líder actual de la liga</name>"
echo "<type>async_string</type>"
echo
"<data><![CDATA[....]]></data>"

// El dato anterior sería generado por un dispositivo/aplicación
dando imágenes en base64.

echo "</module>"
```

Сохраните это содержимое в архив в агенте (или распространите его с коллекциями файлов) и запустите его следующим образом:

```
module_plugin <полный путь к файлу>
```

Специфический мониторинг для Windows

Программный агент для систем Windows имеет специфические функции для этой

платформы, которые помогают облегчить мониторинг. Далее мы объясним функциональные возможности и применим некоторые их примеры. Общие правила:

Если имя процесса содержит пробелы **не используйте** «« «». Имя процесса должно быть таким же, как отображается в диспетчере задач (*taskmngr*) Windows, включая расширение .exe; важно **соблюдать прописные и строчные буквы**.

Мониторинг процессов и watchdog процессов

Мониторинг процессов

Параметр **module_proc** проверяет, работает ли заданное имя процесса на этом устройстве. Пример определения модуля:

```
module_begin
 module_name CMDProcess
 module_type generic_proc
 module_proc cmd.exe
 module_description Process Command line
 module_end
```

Для того чтобы немедленно предупреждать о прекращении работы процесса, необходимо добавить параметр **module_async yes** (см. общие правила в начале раздела Windows):

```
module_begin
 module_name CMDProcess
 module_type generic_proc
 module_proc cmd.exe
 module_async yes
 module_description Process Command line
 module_end
```

Сторожевой таймер Watchdog процессов

Функциональность сторожевого таймера Программного Агента в Pandora FMS для MS Windows® позволяет в случае внезапного завершения процесса действовать немедленно, запуская его заново.

Пример:

```
module_begin
 module_name Notepad
 module_type generic_data
 module_proc notepad.exe
 module_description Notepad
 module_async yes
 module_watchdog yes
 module_user_session yes
 module_start_command "%SystemRoot%\notepad.exe"
 module_startdelay 3000
 module_retrydelay 2000
 module_retries 5
 module_end
```

Каждый раз, когда программа блокнот перестает работать, сторожевой таймер будет выполнять команду:

```
%SystemRoot%\notepad.exe
```

(см. общие правила в начале раздела Windows). Повторная активация процесса предпринимается 5 раз с начальным временем ожидания в 3 секунды и временем ожидания между повторными попытками в 2 секунды в активной сессии пользователя.

Мониторинг сервисов и сторожевой таймер служб

Мониторинг сервисов

Параметр **module_service** проверяет, запущена ли на устройстве определенная служба. Определение модуля с использованием этого параметра будет:

```
module_begin
 module_name Service_Dhcp
 module_type generic_proc
 module_service Dhcp
 module_description Service DHCP Client
 module_end
```

Для того чтобы немедленно предупреждать о прекращении работы процесса, необходимо добавить параметр **module_async yes** (см. общие правила в начале раздела Windows):

```
module_begin
 module_name Service_Dhcp
 module_type generic_proc
 module_service Dhcp
 module_description Service DHCP Client
 module_async yes
 module_end
```

Сторожевой таймер служб

Он работает аналогично Сторожевому таймеру процессов. Пример:

```
module_begin
 module_name ServiceSched
 module_type generic_proc
 module_service Schedule
 module_description Service Task scheduler
 module_async yes
 module_watchdog yes
 module_end
```

Определение сторожевого таймера для служб не требует дополнительных параметров, как, например, для процессов, поскольку эта информация уже содержится в определении службы.

Мониторинг основных ресурсов

В этом разделе показано, как контролировать основные ресурсы устройства Windows.

Мониторинг ЦП

Параметр **module_cpuusage** возвращает процент используемого процессора. Можно отслеживать ЦП по id с помощью следующего определения модуля:

```
module_begin
 module_name CPU_1
 module_type generic_data
 module_cpuusage 1
 module_description CPU usage for CPU 1
 module_end
```

С помощью модуля можно также отслеживать среднее значение использования всех процессоров в системе:

```
module_begin
 module_name CPU Usage
 module_type generic_data
 module_cpuusage all
 module_description CPU Usage for all system
 module_end
```

Мониторинг памяти

Для мониторинга памяти есть два параметра: **module_freememory**, который возвращает количество свободной системной памяти, и **module_freepercentmemory**, который возвращает процент свободной системной памяти.

Пример модуля для **module_freememory**:

```
module_begin
 module_name FreeMemory
 module_type generic_data
 module_freememory
 module_description Non-used memory on system
 module_end
```

Пример модуля для **module_freepercentmemory**::

```
module_begin
 module_name FreePercentMemory
 module_type generic_data
 module_freepercentmemory
 module_end
```

Мониторинг диска

Для мониторинга диска у нас есть два параметра: **module_freedisk**, который возвращает количество свободного места, и **module_freepercentdisk**, который возвращает процент свободного места. Оба модуля требуют в качестве параметра дисковый накопитель для мониторинга, **не забудьте добавить символ :** , например:

```
module_begin
 module_name FreeDisk
 module_type generic_data
 module_freedisk C:
 module_end
```

Пример модуля для **module_freepercentdisk**:

```
module_begin
 module_name FreePercentDisk
 module_type generic_data
 module_freepercentdisk C:
 module_end
```

Запросы WMI

Программный агент Pandora позволяет извлекать информацию с помощью запросов WMI, который является широко используемым источником информации для системной и внешней информации.

Используя параметр **module_wmiquery** Software Agent позволяет локально выполнить любой запрос WMI. Для выполнения запроса вы определяете WMI-запрос в параметре **module_wmiquery** и столбец, содержащий информацию для мониторинга, в параметре **module_wmicolumn**.

Например, получить список установленных служб:

```
module_begin
 module_name Services
 module_type generic_data_string
 module_wmiquery Select Name from Win32_Service
 module_wmicolumn Name
 module_end
```

Получение текущей загрузки процессора:

```
module_begin
 module_name CPU_Load
 module_type generic_data
 module_wmiquery SELECT LoadPercentage FROM Win32_Processor
 module_wmicolumn LoadPercentage
 module_end
```

Версии до 7 NG

Имя агентов

Начиная с версии 7 Pandora FMS, агенты имеют алиас и имя (или уникальный идентификатор). Настроенный по умолчанию агент будет генерировать имя (или идентификатор) на основе псевдослучайной шестнадцатеричной строки, а алиас (или отображаемое имя) - на основе имени хоста устройства.

В предыдущих версиях было только «имя устройства», и предыдущая система полностью совместима с самыми современными версиями Pandora FMS, только если в одной установке Pandora FMS вы найдете двух агентов с одинаковым идентификатором (или именами), данные обоих агентов будут смешиваться и/или накладываться. Именно поэтому, начиная с версии 7, мы ввели возможность того, что агенты с разными именами могут иметь один и тот же алиас.

Для изменения этого поведения используются следующие конфигурационные маркеры:

```
pandora_agent
pandora_alias
```

По умолчанию конфигурационный файл не использует ни то, ни другое, поэтому он получает имя хоста устройства в качестве алиас и очень большое случайное шестнадцатеричное число в качестве имени или идентификатора. Имя агента больше не отображается (кроме как в подробном представлении агента), и его НЕЛЬЗЯ изменить. Алиас агента можно изменить в любое время, не заботясь о конфигурации программного агента, поскольку для уникальной идентификации агента используется»имя» агента.

Вернуться в оглавление Pandora FMS

From:
https://pandorafms.com/manual/!current/ - **current**

Permanent link:
https://pandorafms.com/manual/!current/ru/documentation/03_monitoring/02_operations

Last update: **2024/03/18 21:07**