Структура Pandora FMS

Структура Pandora FMS

Вернуться в оглавление Документации Pandora FMS

Особенности строения Pandora FMS

В следующем разделе мы расскажем о некоторых принципах проектирования и особенностях Pandora FMS.

Дизайн базы данных Pandora FMS

Первые версии Pandora FMS, с 0.83 по 1.1, были основаны на очень простой идее: одна часть данных, одна вставка базы данных. Такая структура позволяла программе быстро выполнять простой поиск, вставку и другие операции.

Помимо всех преимуществ этой разработки, был один недостаток: масштабируемость. Эта система имеет определенное ограничение на максимальное количество модулей, которое она может поддерживать, и при определенном количестве данных (например, 5 миллионов элементов) производительность снижается.

Кластерные решения на основе MySQL, с другой стороны, не так просты: хотя они и могут справиться с большей нагрузкой, они добавляют некоторые дополнительные проблемы и трудности, а также не предлагают долгосрочного решения проблемы производительности при работе с большими объемами данных.

В настоящее время Pandora FMS реализует уплотнение данных в реальном времени для каждой вставки, в дополнение к выполнению сжатия данных на основе интерполяции. С другой стороны, задача обслуживания позволяет автоматически удалять данные, достигшие определенного возраста.

Система обработки Pandora FMS хранит только «новые» данные: если в систему попадает дублирующее значение, оно не сохраняется в базе данных. Это очень полезно, чтобы сохранить базу данных небольшой, и работает для всех типов модулей Pandora FMS (числовых, инкрементных, булевых и строковых). Например, для *булевых* данных индекс компактности высок, поскольку данные трудно изменить. Однако «индексные» элементы сохраняются каждые 24 часа, поэтому при уплотнении информации имеется минимальное количество информации, которая может служить в качестве ссылки.

Этот механизм частично решает проблему масштабируемости, значительно снижая использование базы данных (на 40-70 процентов), но есть и другие способы увеличить масштабируемость.

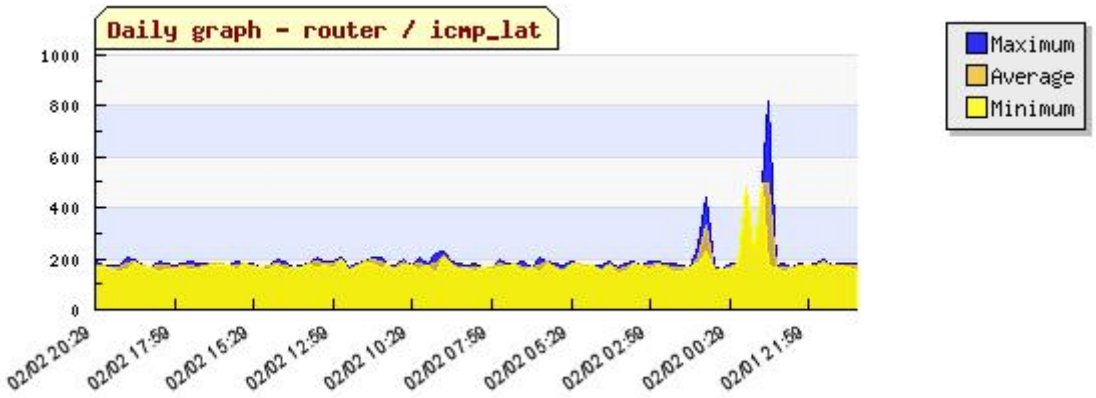
Pandora FMS позволяет полностью развязать компоненты, благодаря чему можно сбалансировать нагрузку по обработке файлов данных и выполнению сетевых модулей на разных серверах. Можно иметь несколько серверов Pandora FMS (сетевые, серверы данных или SNMP-серверы), веб-консоли Pandora FMS, а также базу данных или высокопроизводительный кластер (с MySQL5), и все это на независимых серверах.

Модификации приводят к серьезным изменениям в чтении и интерпретации данных. В последних версиях Pandora FMS графический мотор был переработан с нуля, чтобы иметь возможность быстро представлять данные с помощью новой модели хранения данных.

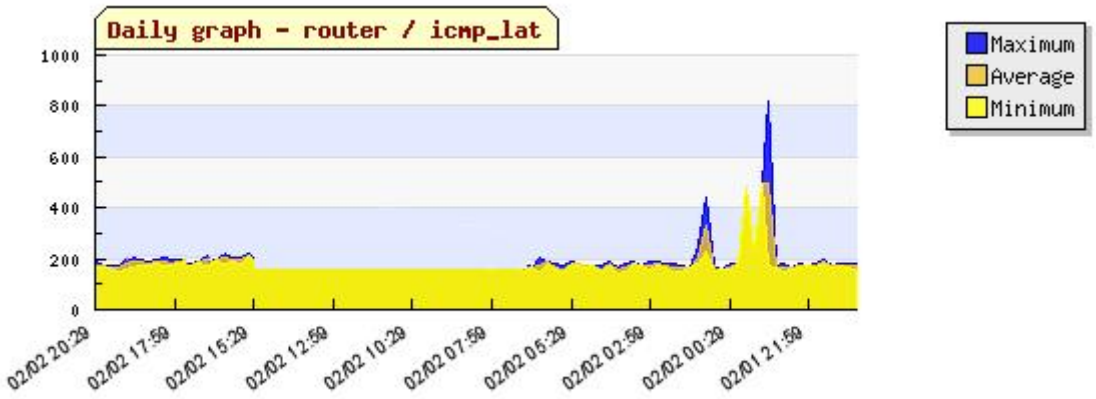
Механизмы уплотнения имеют определенные последствия для чтения и интерпретации данных в графическом виде. Представим, что агент не может связаться с Pandora FMS,

поэтому сервер Pandora FMS не получает от него данные, и будет период времени, в течение которого сервер не имеет информации от модулей этого агента. Если мы обратимся к графику одного из этих модулей, то интервал без данных будет представлен так, как будто изменений не было; горизонтальная линия. Pandora FMS, если она не получает новых значений, считает, что их нет, и все будет выглядеть так, как было в последнем уведомлении.

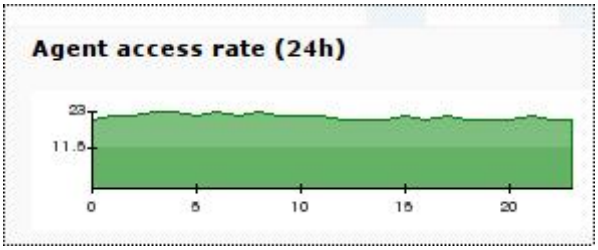
Для наглядного примера на этом изображении показаны изменения для каждых данных, получаемых каждые 180 секунд.



Это эквивалентный график для тех же данных, за исключением сбоя соединения, примерно с 05:55 до 15:29.



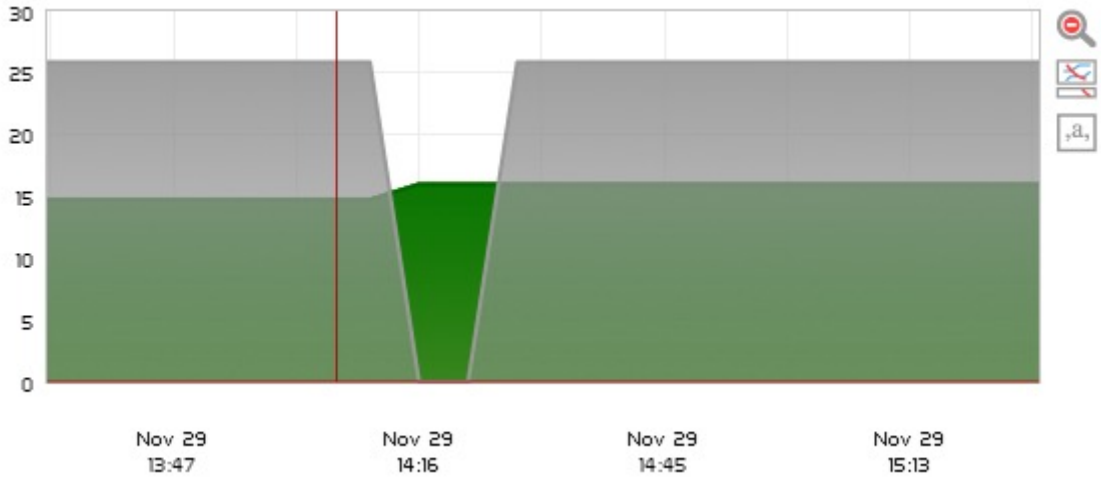
В Pandora FMS 1.3 для агента был введен новый общий график, который показывает связность агента и отражает скорость доступа от модулей к агенту. Этот график дополняет другие, показывающие, когда агент был активен и получал данные.

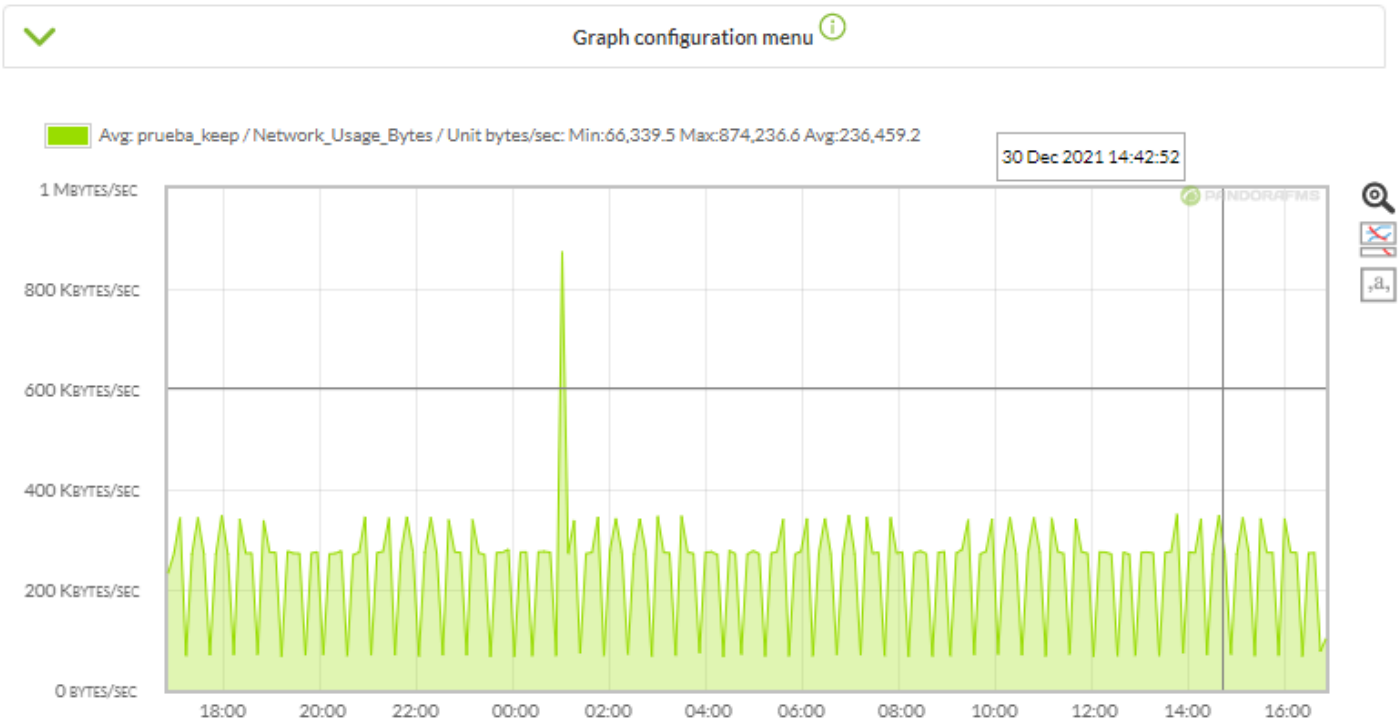


Пример агента, который регулярно подключается к серверу

Если на этом графике есть пики (низкий уровень), у вас могут быть проблемы или медленное соединение в соединении агента Pandora FMS с сервером Pandora FMS, или проблемы с соединением от сетевого сервера.

В Pandora FMS версии 5 была введена возможность пересечения данных событий типа «неизвестный модуль» с графиками, чтобы показать на графике часть серии данных, которая находится под условием неизвестности, дополняя график для лучшей интерпретации, например:





Другие технические аспекты БД

В ходе обновления программного обеспечения были внесены улучшения в реляционную модель базы данных Pandora FMS. Одним из введенных изменений стало индексирование информации на основе различных типов модулей. Таким образом, Pandora FMS может гораздо быстрее получить доступ к информации, поскольку она распределена по разным таблицам.

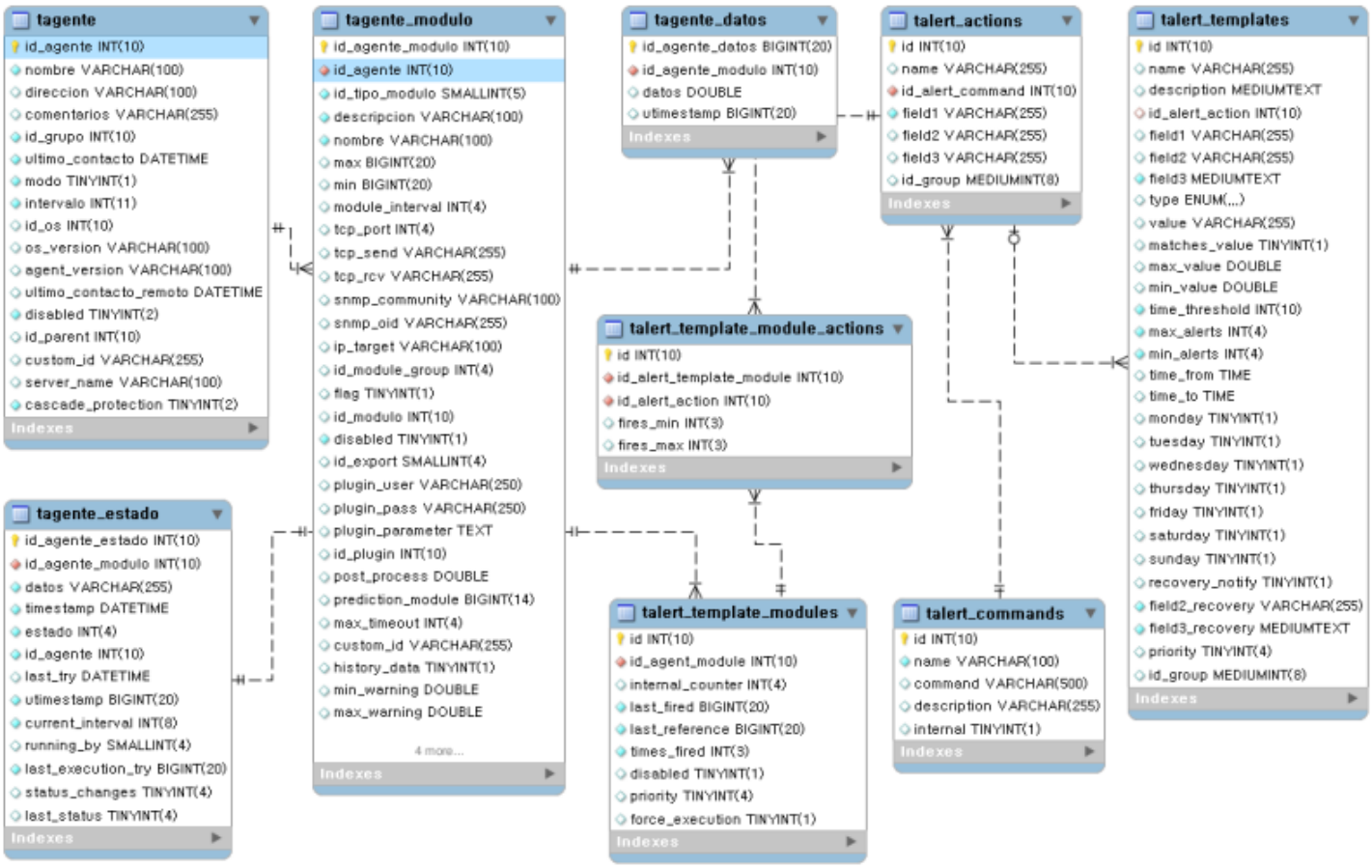
Можно разделить таблицы (по временным меткам) для дальнейшего улучшения производительности доступа к историческим данным.

Кроме того, такие факторы, как числовое представление временных меток (в формате _timestamp_ UNIX), ускоряют поиск диапазонов дат, сравнение дат и т.д. Эта работа привела к значительному улучшению времени поиска и вставки.

Основные таблицы базы данных

Вы можете получить дополнительную информацию о структуре базы данных Pandora FMS (по состоянию на 23 декабря 2020 года) по этой ссылке.

Ниже приведена ER-диаграмма и подробное описание основных таблиц базы данных Pandora FMS.



- taddress: Содержит дополнительные адреса Агентов.
- taddress_agent: Адреса, связанные с агентом (rel. taddress/tagente).
- tagente: Содержит информацию об агентах Pandora FMS.
    - *id_agent*: Уникальный идентификатор агента.
    - *name*: Имя агента (чувствительный к регистру).
    - *address*: Расположение агента. Дополнительные адреса могут быть назначены с помощью таблицы taddress.
    - *комментарии*: Свободный текст.
    - *id_group*: Идентификатор группы, к которой принадлежит агент (ref. tgrupo).
    - *last_contact*: Дата последнего контакта с агентом, либо через программного агента, либо через удаленный модуль.
    - *mode*: Режим, в котором работает агент: 0 - нормальный, 1 - обучение.
    - *interval*: Интервал выполнения агента. В зависимости от этого интервала агент будет помечен как выходящий за пределы.
    - *id_os*: Идентификатор ОС. агента (ref. tconfig_os).

    - *os_version*: Версия ОС. (свободный текст).
    - *agent_version*: Версия агента (свободный текст). Обновляется программными агентами.
    - *last_remote_contact*: Последняя дата контакта, полученная от агента. В случае программных агентов и в отличие от last_contact, дата отправляется самим агентом.
    - *disabled*: Статус агента, включен (0) или отключен (1).
    - *remote*: Агенты с (1) или без удаленной конфигурации (0).
    - *id_parent*: Идентификатор родителя агента (ref. tagente).
    - *custom_id*: Персонифицированный идентификатор агента. Полезен для взаимодействия с другими инструментами.
    - *server_name*: Имя сервера, на который назначен агент.
    - *cascade_protection*: Каскадная защита. Отключена на 0. Установка значения 1 предотвращает срабатывание предупреждений, связанных с агентом, если сработали критические предупреждения от родителя агента. Для получения дополнительной информации, пожалуйста, обратитесь к главе Предупреждения.
    - *safe_mode_module*: ID модуля (того же агента), который использует *safe operation mode* (все остальные модули агента отключаются, если этот модуль находится в критическом состоянии).
- tagent_data: Данные, полученные от каждого модуля. Если для одного и того же Модуля последние полученные данные равны непосредственно предшествующим, то они не вставляются (но *status_tagment* обновляется). Инкрементные и строковые данные хранятся в разных таблицах.
- tagente_datos_inc: Инкрементные данные.
- tagent_data_string: Строковые данные.
- tagente_estado: Информация о текущем состоянии каждого модуля.
    - *id_agent_status*: Идентификатор.
    - *id_agent_module:*: Идентификатор модуля (*ref. tagent_module*).
    - *data*: Значение последних полученных данных.
    - *timestamp*: Дата последних полученных данных (может поступать от агента).
    - *status*: Статус модуля: 0 NORMAL, 1 CRITICAL, 2 WARNING, 3 UNKNOWN.
    - *id_agent*: Идентификатор агента, связанного с модулем (ref. tagent).
    - *last_try*: Дата последнего успешного выполнения модуля.
    - *utimestamp*: Дата последнего выполнения модуля в формате UNIX.
    - *current_interval*: Интервал выполнения модуля в секундах.
    - *running_by*: Имя сервера, на котором запущен модуль.
    - *last_execution_try*: Дата последней попытки выполнения модуля. Исполнение могло быть неудачным.
    - *status_changes*: Количество произошедших изменений состояния. Он используется для того, чтобы избежать непрерывного изменения состояния. Чтобы узнать больше, перейдите в раздел Эксплуатация.
    - *last_status*: Предыдущее состояние модуля.
- tagent_module: Конфигурация модуля.
    - *id_agent_module*: Уникальный идентификатор модуля.
    - *id_agent*: Идентификатор агента, связанного с модулем (ref. tagente).
    - *id_type_module*: Тип модуля (*ref. ttype_module*).
    - *description*: Свободный текст.
    - *name*: Имя модуля.
    - *max*: Максимальное значение модуля. Данные, превышающие это значение, будут считаться недействительными.
    - *min*: Минимальное значение модуля. Данные меньше этого значения будут считаться недействительными.
    - *module_interval*: Интервал выполнения модуля в секундах.
    - *tcp_port*: Порт назначения TCP в сетевых модулях и плагинах. Имя столбца для чтения в

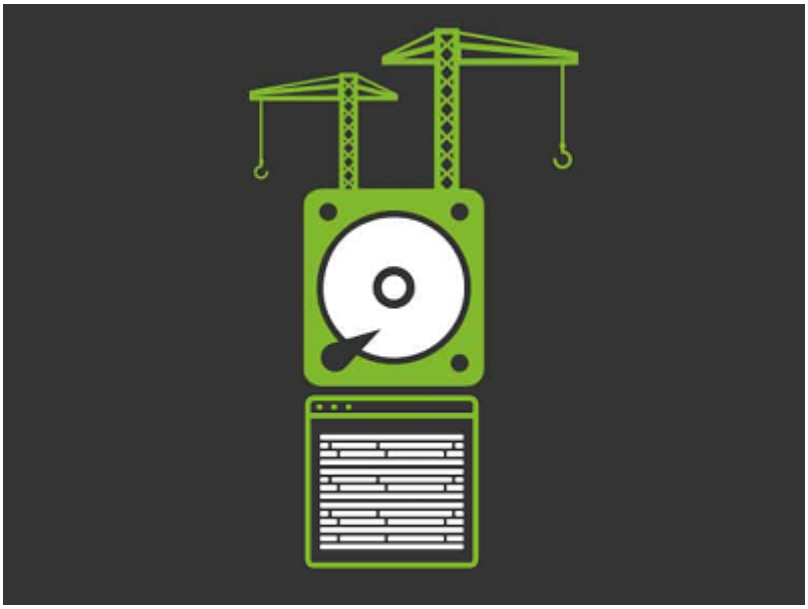
модулях WMI.
    - *tcp_send*: Данные для отправки в сетевых модулях. Пространство имен в модулях WMI.
    - *tcp_rcv*: Ожидаемый ответ в сетевых модулях.
    - *snmp_community*: SNMP сообщество в сетевых модулях. Фильтр в модулях WMI.
    - *snmp_oid*: OID в сетевых модулях. Запросы WQL в модулях WMI.
    - *ip_target*: Адрес назначения в модулях сети, plugin и WMI.
    - *id_module_group*: Идентификатор группы, к которой принадлежит модуль (*ref. tmodule_group*).
    - *flag*: Индикатор или флаг принудительного запуска. Если он равен 1, то модуль выполняется, даже если он не соответствует ему по интервалу.
    - *id_module*: Идентификатор для модулей, которые не могут быть распознаны по их *id_type_module*. 6 для модулей WMI, 7 для WEB-модулей.
    - *disabled*: Состояние модуля, 0 включено, 1 выключено.
    - *id_export*: Идентификатор сервера экспорта, связанного с модулем (ref. tserver).
    - *plugin_user*: Имя пользователя в модулях Plugin и WMI, user-agent в веб-модулях.
    - *plugin_pass*: Пароль в модулях Plugin и WMI, количество повторных попыток в веб-модулях.
    - *plugin_parameter*: Дополнительные параметры в модулях Plugin, конфигурация задачи Goliat в Web-модулях.
    - *id_plugin*: Идентификатор плагина, связанного с модулем в Модули плагинов (*ref. tplugin*).
    - *post_process*: Значение, на которое должны быть умножены данные модуля перед сохранением.
    - *prediction_module*: 1, если это модуль прогнозирования, 2, если это модуль обслуживания, 3, если это синтетический модуль, 0 в любом другом случае./li>
    - *max_timeout*: Время ожидания секунд в модулях *plugin*.
    - *custom_id*: Индивидуальный идентификатор модуля. Полезен для взаимодействия с другими инструментами.
    - *history_data*: Если установлено значение 0, никакие данные модуля не сохраняются в *tagent_data**, обновляется только *tagent_status*.
    - *min_warning*: Минимальное значение, при котором срабатывает статус WARNING.
    - *max_warning*: Максимальное значение, при котором активируется статус WARNING.
    - *min_critical*: Минимальное значение, при котором срабатывает статус CRITICAL.
    - *max_critical*: Максимальное значение, при котором активируется статус CRITICAL.
    - *min_ff_event*: Количество раз, когда условие для изменения состояния должно наступить, прежде чем произойдет изменение состояния. Это связано с tagent_status.status_changes.
    - *delete_pending*: Если установлено значение 1, модуль будет удален скриптом обслуживания базы данных pandora_db.pl в версии Community и pandora_db в версии Enterprise.
    - *custom_integer_1*: Когда *predicion_module = 1*, это поле имеет id модуля, из которого берутся данные для предсказания. Когда *prediction_module = 2*, это поле должно содержать идентификатор службы, назначенной модулю./li>
    - *custom_integer_2*: Используется сервером прогнозирования Prediction Server.
    - *custom_string_1*: Используется сервером прогнозирования Prediction Server.
    - *custom_string_2*: Используется сервером прогнозирования Prediction Server.
    - *custom_string_3*: Используется сервером прогнозирования Prediction Server.
- tagent_access: Новая запись вставляется каждый раз, когда данные поступают от агента на любой из серверов, но не более одной в минуту, чтобы не перегружать базу данных. Можно отключить, установив agentaccess в 0 в конфигурационном файле *pandora_server.conf*.
- talert_snmp: Конфигурация предупреждений SNMP.
- talert_commands: Команды, которые могут быть выполнены из действий, связанных с оповещением (например, отправить электронное письмо).
- talert_actions: Экземпляр команды, связанной с оповещением (например, отправить

электронное письмо администратору).

- talert_templates: Шаблоны предупреждений.
  - *id*: Уникальный идентификатор шаблона.
  - *name*: Имя шаблона.
  - *description*: Описание.
  - *id_alert_action*: Идентификатор действия по умолчанию, связанного с шаблоном.
  - *field1*: Пользовательское поле 1 (свободный текст).
  - *field2*: Пользовательское поле 2 (свободный текст).
  - *field3*: Пользовательское поле 3 (свободный текст).
  - *type*: Тип оповещения в зависимости от условий срабатывания ('regex', 'max_min', 'max', 'min', 'equal', 'not_equal', 'warning', 'critical').
  - *value*: Значение для предупреждений типа regex (свободный текст).
  - *matches_value*: Значение 1 изменяет логику условия срабатывания.
  - *max_value*: Максимальное значение для предупреждений *max_min* и *max*.
  - *min_value*: Минимальное значение для предупреждений *max_min* и *min*.
  - *time_threshold*: Интервал предупреждения.
  - *max_alerts*: Максимальное количество раз, которое должно срабатывать предупреждение в течение интервала.
  - *min_alerts*: Минимальное количество раз срабатывания условия в течение интервала, чтобы предупреждение было запущено.
  - *time_from*: Время, с которого активно предупреждение.
  - *time_to*: Время, до которого активно предупреждение.
  - *monday*: Если 1, предупреждение активно по понедельникам.
  - *tuesday*: Если 1, предупреждение активно по вторникам.
  - *wednesday*: Если 1, предупреждение активно по средам.
  - *thursday*:Если 1, предупреждение активно по четвергам.
  - *friday*: Если 1, предупреждение активно по пятницам.
  - *saturday*: Если 1, предупреждение активно по субботам.
  - *sunday*: Если 1, предупреждение активно по воскресеньям.
  - *recovery_notify*: Если 1, активирует получение предупреждений.
  - *field2_recovery*: Пользовательское поле 2 для получения предупреждений (свободный текст).
  - *field3_recovery*: Пользовательское поле 3 для получения предупреждений (свободный текст).
  - *priority*: Критичность предупреждения: 0 Обслуживание, 1 Информационный, 2 Нормальный, 3 Предупреждение, 4 Критический.
  - *id_group*: Идентификатор группы, к которой принадлежит шаблон (*ref. tgrupo*).
- talert_template_modules: Экземпляр шаблона оповещения, связанный с модулем.
  - *id*: Уникальный идентификатор предупреждения.
  - *id_agent_module*: Идентификатор модуля, связанный с оповещением (*ref. tagent_module*).
  - *id_alert_template*: Идентификатор шаблона, связанного с оповещением (*ref. talert_templates*).
  - *internal_counter*: Количество раз, когда возникло условие срабатывания предупреждения.
  - *last_fired*: Последний случай срабатывания предупреждения (время UNIX).
  - *last_reference*: Начало текущего интервала (время UNIX).
  - *times_fired*: Количество раз срабатывания предупреждения (может отличаться от *internal_counter*).
  - *disabled*: Если 1, предупреждение отключено.
  - *priority*: Критичность предупреждения: 0 Maintenance, 1 Informational, 2 Normal, 3 Warning, 4 Critical.
  - *force_execution*: Если 1, действие предупреждения будет выполнено, даже если оно не было вызвано. Он используется для ручного выполнения предупреждений.

- talert_template_module_actions: Экземпляр действия, связанного с предупреждением. (*ref. talert_template_modules*).
- talert_compound: Составные оповещения, столбцы аналогичны *talert_templates*.
- talert_compound_elements: Простые оповещения, связанные с составным оповещением, каждое из которых имеет соответствующую логическую операцию (*ref. talert_template_modules*).
- talert_compound_actions: Действия, связанные с составным оповещением (*ref. talert_compound*).
- tattachment: Вложения, связанные с инцидентом.
- tconfig: Конфигурация консоли.
- tconfig_os: Операционные системы, распознаваемые в Pandora FMS.
- tevento: Записи о событиях. Значения приоритета такие же, как и у оповещений.
- tgrupo: Группы, определенные в Pandora FMS.
- tincidencia: Записи происшествий.
- tlanguage: Языки, распознаваемые в Pandora FMS.
- tlink: Ссылки, отображаемые в нижней части меню консоли.
- tnetwork_component: Сетевые компоненты. Это модули, связанные с сетевым профилем, используемым Recon Server. Впоследствии это приведет к записи в *tagent_module*, так что столбцы обеих таблиц будут похожи.
- tnetwork_component_group: Группы для классификации компонентов сети.
- tnetwork_profile: Профиль сети. Группировка сетевых компонентов, которым назначены задачи распознавания Recon Server. Сетевые компоненты, связанные с профилем, будут порождать модули в созданных Агентах.
- tnetwork_profile_component: Сетевые компоненты, связанные с сетевым профилем (*rel. tnetwork_component/tnetwork_profile*).
- tnota: Примечания, связанные с инцидентом.
- torigen: Возможные причины возникновения.
- tperfil: Профили пользователей, определенные в консоли.
- trecon_task: Разведывательные задачи Recon Server.
- tserver: Зарегистрированные серверы.
- tsesion: Информация о действиях, выполненных во время сеанса пользователя, для журналов администрирования и статистики.
- ttipo_modulo: Типы модулей в соответствии с их происхождением и типом данных.
- ttrap: SNMP-ловушки, полученные SNMP-консолью.
- tuser: Пользователи, зарегистрированные в Консоли.
- tuser_profile: Профили, связанные с пользователем (*rel. tuser/tprofile*).
- tnews: Новости, отображаемые в консоли.
- tgraph:Пользовательские графики, созданные в консоли.
- tgraph_source: Модули, связанные с графиком (*rel. tgraph/tagent_module*).
- treport: Индивидуальные отчеты, созданные в Консоли.
- treport_content: Элементы, связанные с отчетом.
- treport_content_sla_combined: Компоненты элемента SLA, связанного с отчетом.
- tlayout: Пользовательские карты, созданные в Консоли.
- tlayout_data: Элементы, связанные с картой.
- tplugin: Определения плагинов для Plugin Server./li>
- tmodule: Типы модулей (Network, Plugin, WMI...).
- tserver_export: Место назначения, настроенное для Export Server.
- tserver_export_data: Данные для экспорта, связанные с местом назначения.
- tplanned_downtime: Запланированные остановки.
- tplanned_downtime_agents: Агенты, связанные с плановым отключением (*rel. tplanned_downtime/tagente*).

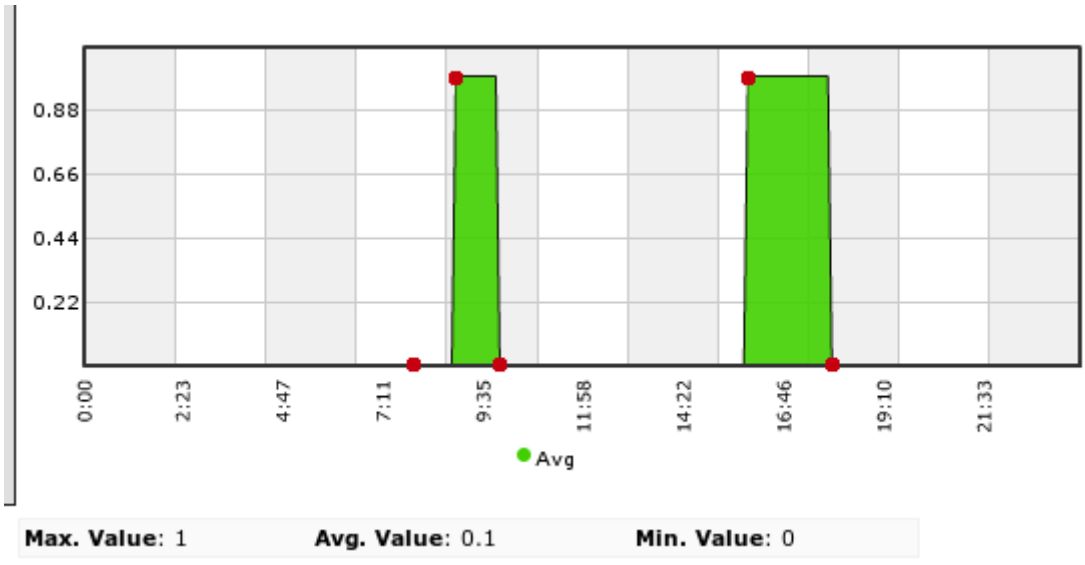
Сжатие данных в режиме реального времени



Как упоминалось выше, чтобы не перегружать базу данных, сервер выполняет простое сжатие во время вставки. Элемент данных не сохраняется в базе данных, *если он не отличается от предыдущего или между ними есть разница в 24 часа*.

Например, предполагая приблизительный интервал в 1 час, последовательность 0,1,0,0,0,0,0,0,1,1,0,0 хранится в базе данных как 0,1,0,1,0. Очередной последовательный 0 сохраняется только по истечении 24 часов.

Приведенный ниже график был построен на основе данных, приведенных в примере выше. В базу данных были вставлены только данные, отмеченные красным цветом.



Сжатие сильно влияет на алгоритмы обработки данных, как механические, так и графические, и важно помнить, что пробелы, вызванные сжатием, должны быть заполнены.

Учитывая все вышесказанное, для выполнения расчетов с данными модуля, заданного интервалом и начальной датой, необходимо выполнить следующие шаги:

- Поиск предыдущих данных без учета заданного интервала и даты. Если они существуют, поместите их в начало диапазона. Если не существует, то данных нет.
- Поиск следующих данных, вне заданного интервала и даты до максимума, равного интервалу модуля. Если они существуют, их следует ввести в конце интервала. Если их нет, то последнее доступное значение должно быть продлено до конца интервала.
- Все данные просматриваются, принимая во внимание, что часть данных действительна до тех пор, пока не будет получена другая часть данных.

Уплотнение данных

В Pandora FMS предусмотрена система «уплотнения» информации в базе данных. Эта система ориентирована на сценарии малого/среднего размера (250-500 агентов, 100 000 модулей), которые хотят иметь богатую историю информации, но при этом не терять разрешения.

Обслуживание базы данных Pandora FMS, которое выполняется каждый час и, помимо прочих задач по очистке, позволяет произвести уплотнение старых данных. Это уплотнение использует простую линейную интерполяцию, что означает, что если, например, у вас есть 10 000 точек данных за день, процесс уменьшает эти 10 000 точек на 1 000 точек.

Поскольку это интерполяция, это приводит к потере детализации этой информации, но она все еще достаточно информативна для создания ежемесячных, ежегодных и т.д. отчетов и графиков.

В больших базах данных такое поведение может быть довольно дорогостоящим с точки зрения производительности, и его придется отключить; вместо этого рекомендуется выбрать историческую модель базы данных.

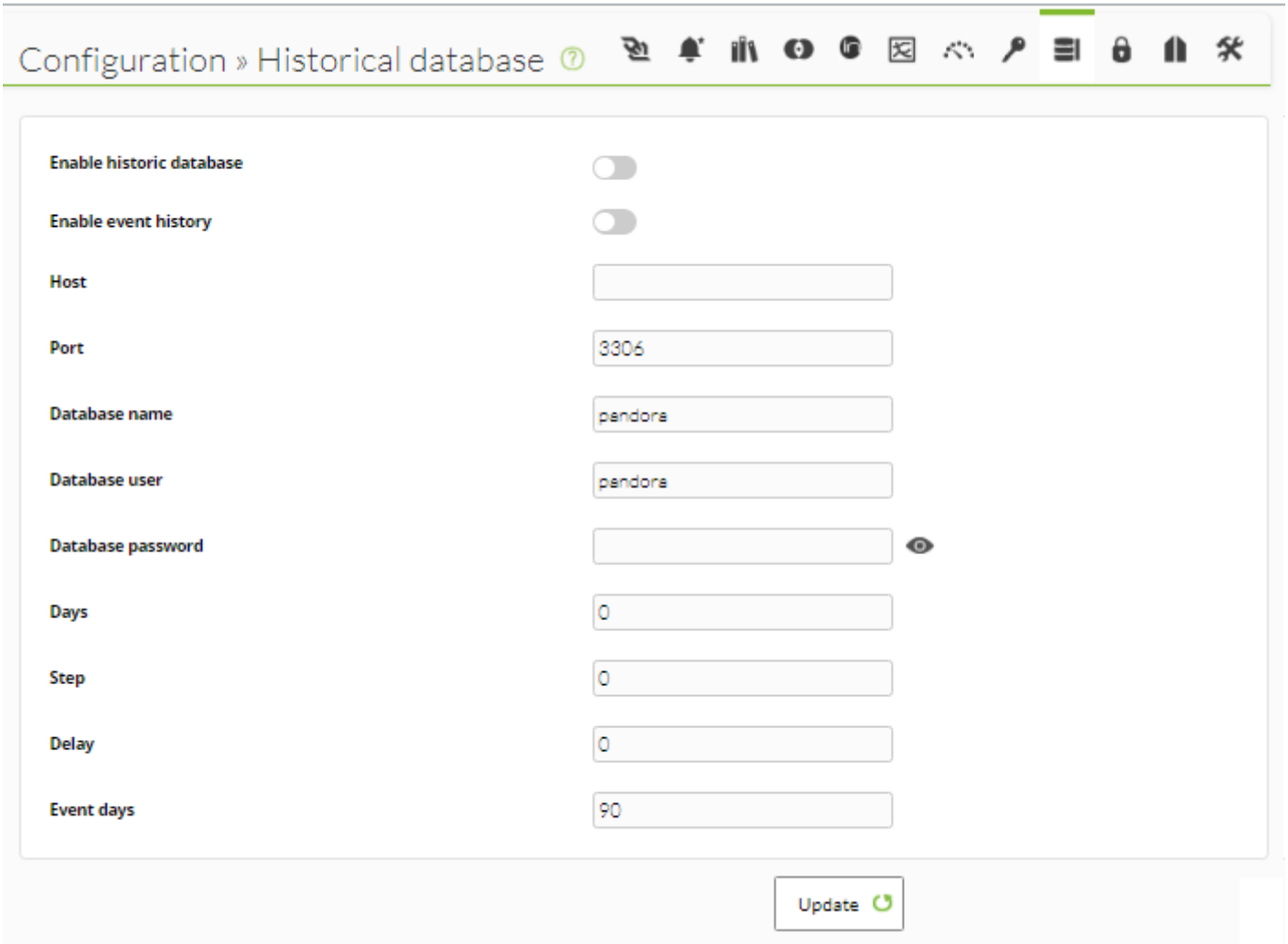
Историческая база данных

Историческая база данных - это функция версии Enterprise, используемая для хранения всей прошлой информации, которая не используется при просмотре последних дней, например, данные старше одного месяца. Эти данные автоматически переносятся в другую базу данных, которая должна находиться на другом сервере с другим хранилищем (диском), чем основная база данных.

Когда показывается график или отчет со старыми данными, Pandora FMS будет искать первые дни в основной базе данных, а когда дойдет до момента перехода в историческую базу данных, будет искать в ней. В результате производительность максимизируется даже при накоплении большого количества информации в системе.

Чтобы настроить это, необходимо вручную установить на другом сервере историческую базу данных (импортировав в нее схему Pandora FMS без данных), а также установить разрешения на доступ к ней с главного сервера Pandora FMS.

Затем перейдите в Setup → Setup → Historical database и настройте параметры для доступа к исторической базе данных.



Некоторые из параметров, которые необходимо принять во внимание, следующие:

Days

Максимальное количество дней, в течение которых информация хранится в основной базе данных. После этой даты данные будут перенесены в историческую базу данных. Тридцать (30) дней - это хороший запас.

Step

Действуя как *буфер*, сценарий обслуживания базы данных будет брать X записей из основной базы данных, вставлять их в историческую базу данных и удалять их из основной базы данных. Это занимает время, а размер зависит от конфигурации консоли. Хорошим значением является одна тысяча (1000).

Delay

После блока серии модулей *script* ожидает в течение секунд, указанных в *delay*. Это полезно, если производительность базы данных низкая, поскольку позволяет избежать сбоев. Значения должны быть от одного до пяти (1–5).

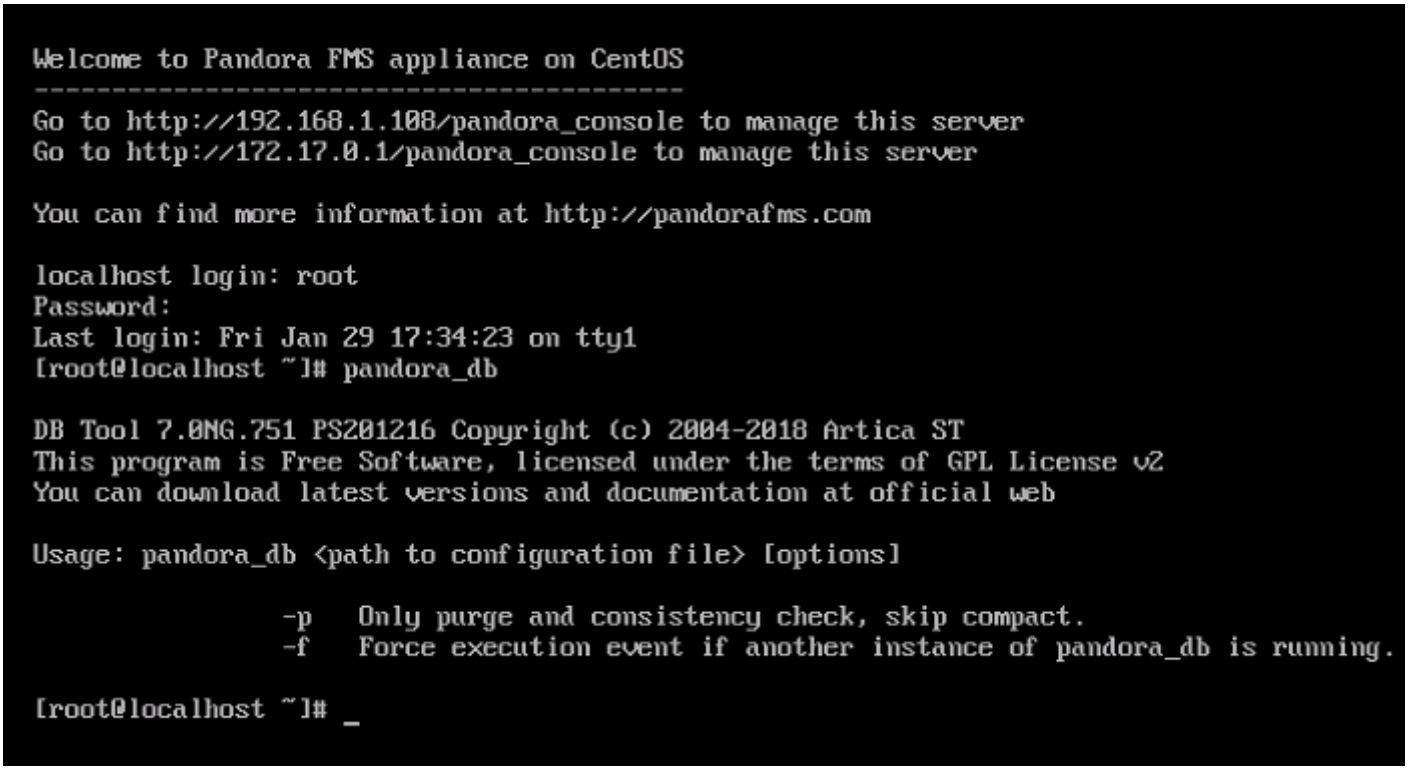
Расширенная конфигурация

Конфигурация Pandora FMS по умолчанию не передает строковые данные в базу данных истории, однако если эта конфигурация была изменена и база данных истории получает подобную информацию, необходимо настроить ее очистку, иначе она будет занимать слишком много места, вызывая серьезные проблемы и негативно влияя на производительность.

Чтобы настроить этот параметр, необходимо выполнить прямой запрос к базе данных, чтобы определить дни, после которых эта информация будет удалена. Речь идет о таблице tconfig и поле string_purge. Чтобы установить 30 дней для очистки такого рода информации, следующий запрос будет выполнен непосредственно в базе данных истории:

```
UPDATE tconfig SET value = 30 WHERE token = "string_purge";
```

База данных поддерживается скриптом под названием pandora_db.pl:



Чтобы проверить, правильно ли выполняется обслуживание базы данных, запустите сценарий *maintenance script* вручную:

```
/usr/share/pandora_server/util/pandora_db.pl /etc/pandora/pandora_server.conf
```

Не должно появиться каких-либо ошибок. Если другой экземпляр использует базу данных, вы вполне можете использовать параметр `-f`, который принудительно выполняет команду; с параметром `-p` он не выполняет уплотнение данных. Это особенно полезно в средах Высокой доступности (HA) с исторической базой данных, поскольку *скрипт* следит за тем, чтобы выполнить в правильном порядке и правильным способом необходимые шаги для этих компонентов.

Состояние модулей Pandora FMS

В Pandora FMS модули могут иметь различные состояния: неизвестное, нормальное, предупреждение, критическое или с включенным оповещением.

Когда определяется каждое состояние?

С одной стороны, каждый модуль имеет в своей конфигурации пороговые значения *Предостережение* у *Критическое*. Эти пороговые значения определяют значения ваших данных, для которых активируются эти состояния. Если модуль предоставляет данные за пределами этих пороговых значений, то считается, что он находится в *нормальном* состоянии.

Каждый модуль также имеет временной интервал, который задает частоту сбора данных. Этот интервал должен быть учтен консолью для сбора данных. Если модуль не собирал данные в течение удвоенного интервала, считается, что модуль находится в состоянии *Неизвестный*.

Наконец, если модуль имеет настроенные оповещения, и любое из них было запущено и не было подтверждено, модуль будет иметь соответствующий статус *Сработало предупреждение*.

Распространение и расстановка приоритетов

В структуре Pandora FMS одни элементы зависят от других, как, например, модули агента или агенты группы. Это также можно применить к случаю политик Pandora FMS Enterprise, которые имеют ассоциированные определенные агенты и определенные модули, которые считаются ассоциированными с каждым агентом.

Эта структура особенно полезна для *оценки состояния модулей с первого взгляда*. Состояния распределяются снизу вверх, таким образом, придавая статус агентам, группам и политикам.

Какой статус будет у агента?

Агент будет показан на основе *наихудшего* состояния из его модулей. В свою очередь, группа будет иметь состояние самого худшего из агентов, входящих в нее, и то же самое для политик, которые будут иметь наихудшее состояние назначенных им агентов.

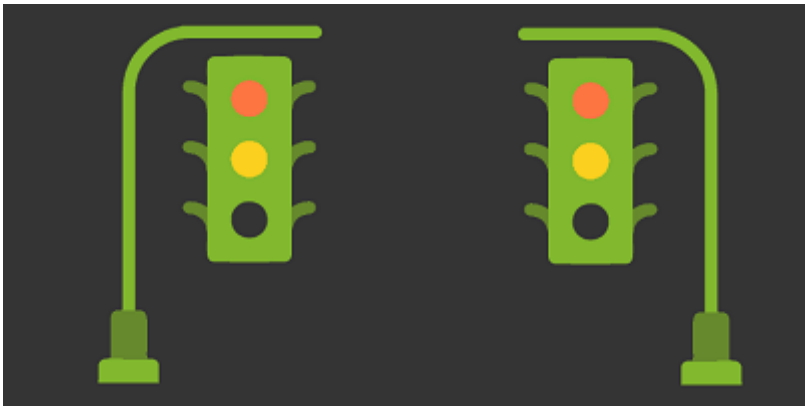
Таким образом, *когда вы видите группу с критическим статусом*, например, вы будете знать, что по крайней мере один из ваших агентов имеет такой же статус. Чтобы найти его, необходимо спуститься еще на один уровень, на уровень агентов, чтобы сузить круг и найти модуль или модули, ответственные за распространение этого критического состояния.

Каков приоритет состояний?

Когда мы говорим о распространении *наихудшего* состояния, мы должны четко понимать, какие состояния имеют приоритет над остальными. Поэтому существует список приоритетов, в котором первое состояние в списке имеет самый высокий приоритет над остальными, а последнее - самый низкий, появляясь только тогда, когда его имеют все элементы.

1. Срабатывание предупреждений.
2. Критическое состояние.
3. Состояние предупреждения.
4. Неизвестное состояние.
5. Нормальное состояние.

Мы видим, что когда модуль вызвал оповещения, его состояние имеет приоритет над всеми остальными, и агент, которому он принадлежит, будет иметь это состояние, и группа, к которой принадлежит этот агент, также будет иметь это состояние.



С другой стороны, чтобы группа, например, имела нормальное состояние, все ее агенты должны иметь нормальный статус, что означает, что все модули этих групп будут иметь нормальный статус.

Цветовой код

Каждому из прокомментированных состояний (статусов) присвоен цвет, чтобы вы могли сразу увидеть, когда элемент не работает на картах сети.

- Состояние сработавших оповещений
- Критическое состояние
- Состояние предостережения
- Неизвестное состояние
- Нормальное состояние

Графики Pandora FMS

Графики являются одним из самых сложных элементов Pandora FMS, они извлекают данные в реальном времени из базы данных, при этом не используется никакая внешняя система (например, RRDtool или аналогичная).

Существует несколько вариантов поведения графиков в зависимости от типа исходных данных:

- Несинхронные модули. Предполагается, что компактность данных отсутствует. Все данные в базе данных представляют собой реальные выборки данных, уплотнение отсутствует. С их помощью создаются гораздо более «точные» графики без возможности неправильного толкования.
- Модули типа текстовой строки. Они показывают графики со скоростью передачи данных во времени.
- Модули числовых данных. Большинство модулей передают такой тип данных.
- Модули булевых данных. Соответствуют числовым данным на мониторах **PROC*: например, проверка Ping, состояние интерфейса и т.д. Значение 0 соответствует критическому состоянию, а значение 1 - «нормальному» состоянию.

Компрессия

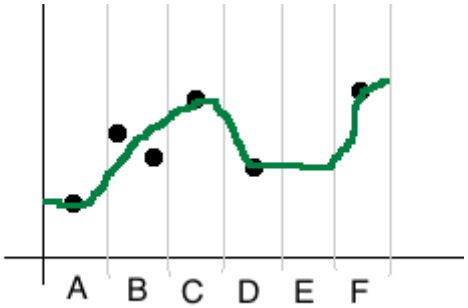
Сжатие влияет на способ изображения графиков. При получении двух данных с одинаковым значением Pandora FMS не сохраняет последнее. Кроме того, в случае, если при создании графика нет эталонного значения, Pandora FMS будет искать последнее известное значение за последние 48 часов, которое она примет за эталон. Если данные не найдены, график начнется с 0.

В несинхронных, несмотря на отсутствие сжатия, механизм обратного поиска будет вести себя аналогичным образом.

Интерполяция

При составлении графика берется 50xN образцов, где N - коэффициент разрешения графиков (этот параметр можно настроить в настройках, по умолчанию он равен 3). Например, прибор контроля, возвращающий данные каждые 300 секунд (5 минут), будет генерировать 12 проб в час, 288 (*12*24*) в день. В случае однодневного графика 288 точек не «печатаются» в реальности, а «сжимаются» с использованием всего 150 (*50*3*) образцов.

Это означает, что «теряется» некоторое разрешение, и чем больше образцов мы имеем, тем больше оно теряется, но этого можно избежать, создав график с другим интервалом или масштабом.

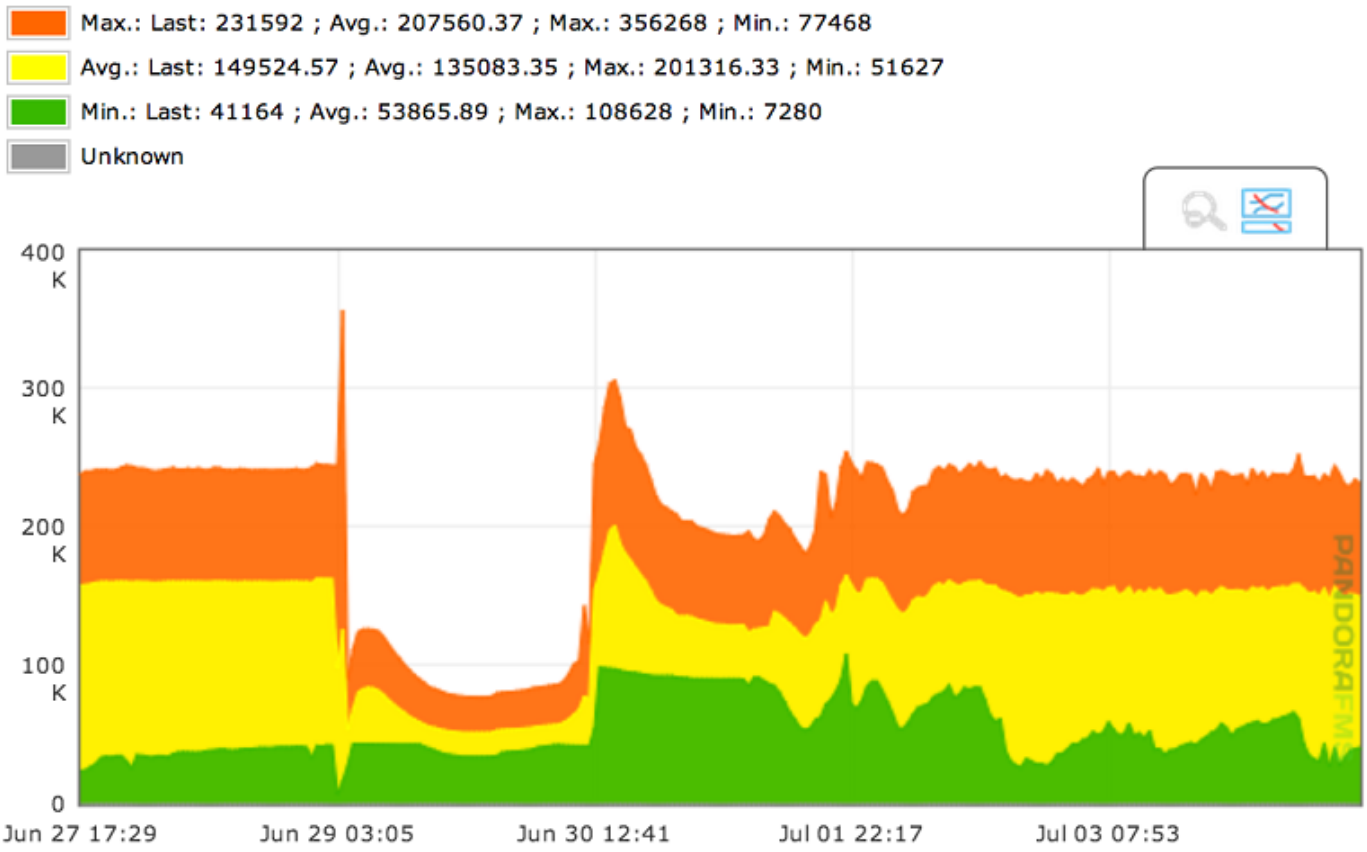


В нормальном графике интерполяция реализуется простым способом: если в интервале есть две выборки (например, в интервале B в примере), среднее значение усредняется, и отображается именно это значение.

В булевых графиках, если в выборке доступно несколько данных (в данном случае могут быть доступны только 0 и 1), отображается 0. Это наглядно помогает увидеть случай отказа в интервале, определяя приоритет проблемы по сравнению с текущей ситуацией.

В обоих случаях, если в выборке отсутствуют какие-либо данные (из-за их сжатия или отсутствия), для отображения используется последнее известное значение предыдущего интервала, как в интервале E в примере выше.

Avg/Max/Min



Графики по умолчанию показывают среднее, максимальное и минимальное значение. Поскольку в выборке может быть несколько данных, отображаются данные для среднего (*avg*), максимального или минимального значения. Чем больше график должен быть интерполирован (чем больше период отображения и чем больше данных), тем больше степень интерполяции и, следовательно, тем больше различий между значениями *max*, *min* и *avg*.

Чем меньше диапазон графика (например, часовые графики), тем меньше будет интерполяция, или она будет очень легкой, так что данные будут видны в их реальном разрешении, и три серии будут идентичны.

Вернуться в Оглавление документации Pandora FMS