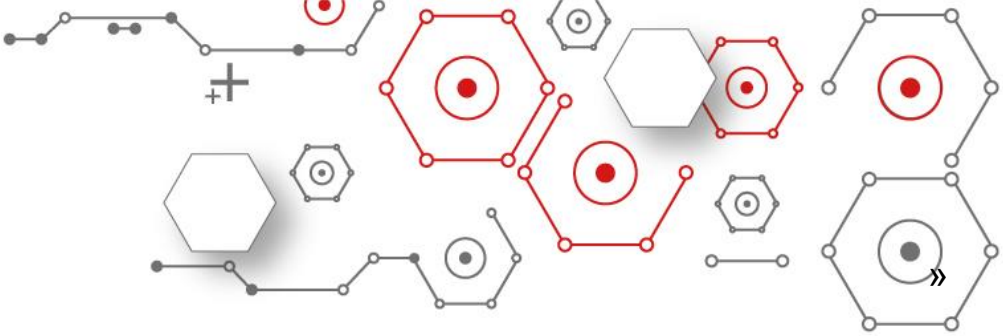


Инструкция по эксплуатации
Программного обеспечения

Система сбора и анализа событий информационной безопасности «SAVA»

ОБЩИЕ СВЕДЕНИЯ

Полное наименование: Система сбора и анализа событий информационной безопасности «SAVA» (далее- «Система»)

Условное обозначение Системы: SAVA.

Задачи, решаемые Системой

1. Обнаружение и сбор событий безопасности, формируемых агентами.
2. Обеспечение хранилища нормализованных событий безопасности.
3. Формирование отчетов и оповещения об инцидентах ИБ.
4. Оперативный контроль за защищенностью подключенных информационных систем.
5. Доступность инструментов анализа собираемой информации.

Особенности системы

Методика работы Системы заключается в нескольких этапах:

- получении отчётов (записей зарегистрированных событий или логов) автоматизированных систем обеспечения информационной безопасности, систем обработки и хранения информации, а также операционных систем;
- приведению записей к общему виду;
- разбору этих записей на токены для классификации, обогащения, обработки и анализа;
- хранение записей как параметризированных объектов;
- сопоставление последовательностей объектов и/или их параметров сценариям корреляции;
- регистрация таких соответствий и оповещение о них.

Основная сложность задач Системы заключается в следующих фактах:

1. Общее число разновидностей форматов записей событий в одной только корпоративной экосистеме может исчисляться более чем 100 форматов. Это вынуждает поддерживать актуальность компонентов сбора информации на постоянной основе и выпускать обновлённые компоненты для Агента Системы;
2. Корпоративная экосистема может состоять как из мене, чем 10 источников событий информационной безопасности, так и более чем 500 000 таковых. Соответственно технические требования для обеспечения необходимой производительности Системы будут иметь динамический характер. А сама Система строится на распределённых компонентах для диверсификации вычислительных ресурсов;
3. Сценарии корреляции формируются Администратором системы, что не исключает человеческого фактора в самом фундаменте системы мониторинга. Малейшая ошибка человека при конфигурировании системы может влиять на эффективность её работы;
4. Установка и пусконаладка Системы требует высокого уровня компетенции системного администратора и наличия дополнительных служб обеспечения хранения, обмена и обработки данных.

Зависимости Системы:

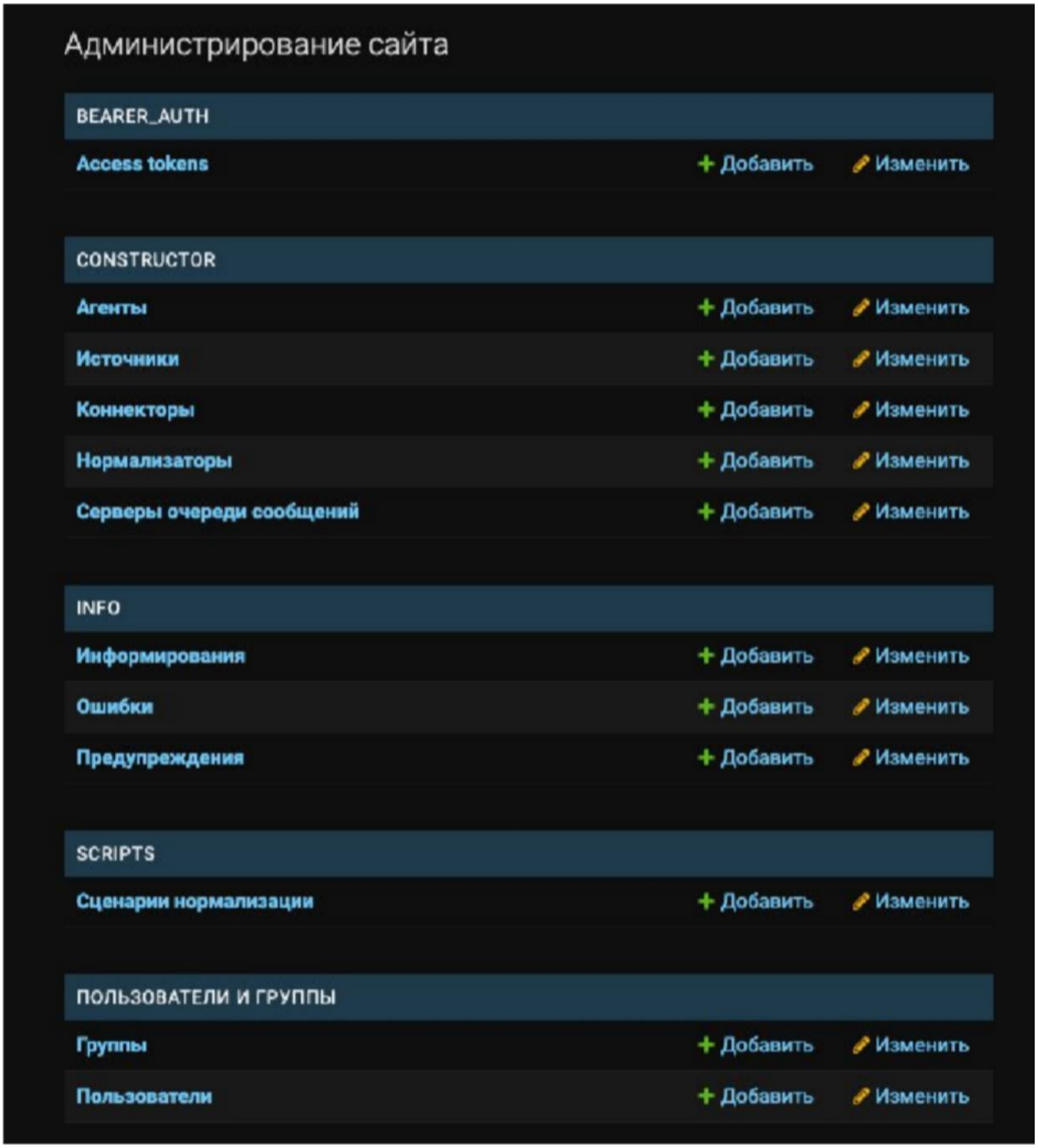
1. Система контейнеризации сервисов (Docker-compose)
2. Диспетчер очередей сообщений (RabbitMQ)
3. Система управления базами данных (Postgresql)
4. Система хранения ключ-значение (Redis)
5. Поисковый движок (Elasticsearch)
6. Интерпретатор языка Python 3.10

Расчёт технических требований к стабильной производительной работе Системы:

- До 10 источников событий - Наиболее критичным ресурсом для Системы является оперативная память. Это первостепенный ресурс, который вероятнее всего закончится первым. Минимальный допустимый размер — 8Gb, рекомендуемый — от 16 до 64 Gb. Допускается больше, если в этом действительно есть необходимость. Выбирать стоит современный процессор с несколькими ядрами. Как правило, сервера в кластере используют двух-восьмиядерные машины. Если стоит выбор между более быстрыми процессорами или процессорами с несколькими ядрами, стоит выбирать последние. Дополнительный параллелизм, предлагаемый несколькими ядрами, даст больший результат, нежели чуть более высокая тактовая частота.
- До 1000 источников событий - Минимальный допустимый размер ОЗУ — 16Gb, рекомендуемый — от 32 до 64 Gb. Серверный кластер из 2 и более машин для разделения ядра Системы и поискового движка
- До 5000 источников событий - Серверный кластер из 4 и более машин для разделения поискового движка на отдельный кластер из 3 машин
- До 30000 источников событий - Серверный кластер из 5 и более машин для разделения поискового движка на отдельный кластер из 3 машин и отдельный сервер для диспетчера очереди сообщений

Конфигурирование Системы

Доступ к панели конфигуратора Системы – http://127.0.0.1:8080/admin
Login: root
Password: Arinteg123!



1. Добавляем все источники событий информационной безопасности в список «Источники раздела Constructor»
2. Добавляем все коннекторы (типы источников) в список «Коннекторы раздела Constructor»
3. Добавляем все Агенты в список «Агенты раздела Constructor»
4. Добавляем сервер очереди сообщений в список «Серверы очереди сообщений раздела Constructor»

5. Создаём необходимое количество экземпляров (процессов) нормализатора записей событий
6. Добавляем сценарий Нормализации событий по примеру:

Как выглядит Json сценария "WinEventLog"


```
[
 {
  "правило": "конвертировать",
  "поле": "raw",
  "формат": "json"
 },
 {
  "правило": "проверка значения",
  "поле": "variable.EventCode",
  "значение": {
   "4634": [
    {
     "правило": "создать поле",
     "поле": "message",
     "содержание": "Пользователь завершил рабочую сессию на устройстве"
    },
    {
     "правило": "регулярное выражение",
     "поле": "variable.Message",
     "выражение": "A.*Subject:?[\\t|\\n|\\r]*Security ID:?[\\t|\\n|\\r]*(.*)?[\\t|\\n|\\r]*Account Name:?[\\t|\\n|\\
r]*(?P<username>.*)?[\\t|\\n|\\r]*Account Domain:?[\\t|\\n|\\r]*(?P<domainname>.*)?[\\t|\\n|\\r]*Logon ID:?[\\t|\\n|\\r]*(.*)?[\\
t|\\n|\\r]*Logon Type:?[\\t|\\n|\\r]*(.*)?[\\t|\\n|\\r]* ?.*"
    },
    {
     "правило": "скопировать значение поля",
     "поле": "variable.CategoryString",
     "скопировать в": "event.type"
    },
    {
     "правило": "скопировать значение поля",
     "поле": "variable.TimeGenerated",
     "скопировать в": "time"
    },
    {
     "правило": "скопировать значение поля",
     "поле": "variable.username",
     "скопировать в": "user.name"
    },
    {
     "правило": "скопировать значение поля",
     "поле": "variable.domainname",
     "скопировать в": "user.domain"
    },
    {
     "правило": "скопировать значение поля",
     "поле": "variable.ComputerName",
     "скопировать в": "device.name"
    }
   ]
  }
 }
]
```


Запуск Агента сбора логов из директории «./components/agent/» командой «python main.py»

Как работает сценарий