GS LABS
GS GROUP

Публичный - L0

DRE Advanced Encryption Service

Руководство администратора

Индекс	2004-DREAdvancedEncryptionService-AG
Секретность	Публичный - L0
Ревизия	1.0
Статус	Согласован
Подразделение	ДПРСУД
Компания	GS Labs

Содержание

1. Аннотация

Документ содержит настройки продукта DRE Advanced Encryption Service (далее - ADECScrambler или Система) и предназначен для сотрудников отдела мониторинга и инсталляции, а также для других технических специалистов, в обязанности которых входит настройка системы и поддержание её работоспособности.

2. Термины и сокращения

Термин	Определение
Транспортный поток (TS)	Набор объединенных элементарных потоков, используемый для передачи аудио, видео и других данных в системах цифрового вещания. Структура транспортного потока определена в стандарте ISO/IEC 13818-1.
Элементарный поток (ES)	Поток данных одного типа, передающийся в составе транспортного потока. Примеры: аудиодорожка, видео, телетекст, служебная информация.
Скремблер (Scrambler)	Устройство шифрования транспортного потока, входящее в состав головного оборудования. В терминологии стандарта DVB-Simulcrypt обозначает функциональный логический блок, ответственный за шифрование MPEG2 транспортного потока. Конкретная функциональность зависит от реализации.

Сокращение	Расшифровка
ADEC	(ADvanced EnCryption) - система шифрования транспортного потока, применяемая в дополнение к стандартному алгоритму шифрования (CSA).
ES	Elementary Stream, Элементарный поток (см. таблицу терминов)
MPEG	(от Moving Picture Experts Group – Группа Экспертов по Движущемуся Изображению) – название системы кодирования набора сжатых цифровых телевизионных видеосигналов, звуковых сигналов и данных пользователя телевизионной информации в поток цифровых пакетов
IP	(Internet Protocol) – протокол передачи данных по сети Интернет
PID	Идентификатор пакетов, относящихся к одному элементарному потоку. Уникален в пределах транспортного потока.
TS	Transport Stream, Транспортный поток (см. таблицу терминов)

3. Введение

3.1. Требования к квалификации администратора

Администратор Системы должен обладать навыками работы с ОС семейства Linux (Debian, Ubuntu) (запуск служб, просмотр и редактирование файлов с помощью текстовых редакторов), файлами в формате *json*, а также навыками работы с Docker и gitlab.

3.2. Обслуживание системы

Обслуживание ADECScrambler заключается в выполнении следующих основных действий:

- Изменение настроек компонентов системы (**при необходимости**). После изменения настроек необходимо перезапустить соответствующий контейнер, чтобы изменения вступили в силу.
- Устранение ошибок в работе ADECScrambler на основе его логов.
- Мониторинг работы системы с помощью метрик.

При установке ADECScrambler с нуля необходимо настроить Систему под нужды конкретного Заказчика (перед первым запуском нужно отредактировать конфигурационные файлы, указав нужные параметры входных и выходных данных). Работа ADECScrambler при дефолтных настройках не гарантируется.

Перейти к Содержанию...

4. Настройка ADECScrambler

4.1. Конфигурационные файлы

Для работы ADECScrambler используются следующие файлы:

- ***adec_scrambler_go.cfg.dft*** - конфигурационный файл ADECScrambler;
- ***pid_map.cfg*** - файл конфигурации каналов;
- файл с ключами для шифрования. Бывает двух видов:
  - ***plain_keys.json*** - пример plain файла с ключами шифрования. Нешифрованный файл, допускает редактирование.
  - шифрованный бинарный файл с ключами. Файл предоставляется лицами, ответственными за ADEC ключи (KeyOwners). Шифрование осуществляется специальной утилитой (предоставляется по запросу заказчика), пароль заносится в конфигурационный файл ADECScrambler. Редактирование файла не требуется.

Описание параметров, задаваемых в конфигурационных файлах, а также примеры этих файлов содержатся в отдельном документе, предоставляемом по запросу заказчика.

Перейти к Содержанию...

5. Управление ADECScrambler

ADECScrambler поставляется в виде образа (ссылка предоставляется по запросу заказчика), из которого разворачивается в Docker-контейнер. В связи с этим управление ADECScrambler-ом = управление Docker-контейнерами.

НЕ РЕКОМЕНДУЕТСЯ осуществлять управление ADECScrambler-ом внутри Docker-контейнера (например, ручной запуск службы с параметрами) без крайней необходимости.

Команды для работы с Docker-контейнерами см. здесь: https://docs.docker.com/engine/reference/commandline/docker/

⚠ **Обратите внимание!** При развертывании ADECScrambler имя контейнера не фиксировано.

Перейти к Содержанию...

6. Ведение Логов

6.1. Просмотр логов

Все логи пишутся в stdout Docker-контейнера, в связи с этим дополнительная запись логов в файл отключена.

Для просмотра логов нужно выполнить команду:

```
docker logs [OPTIONS] CONTAINER
```

где:

- [OPTIONS] - дополнительные параметры при выполнении команды. См. https://docs.docker.com/engine/reference/commandline/logs/
- CONTAINER - название docker-контейнера с сервисом.

6.2. Режимы ведения логов

Компоненты SCRAMBLER ведут логи, информация из которых может быть использована для решения возникающих проблем. Логи могут вестись с разной степенью подробности.

Уровни логирования настраиваются в **adec_scrambler_go.cfg** с помощью параметра LOGGER_LEVEL.

Доступны следующие режимы ведения логов:

- 0 - trace: подробная информация по любым действиям;
- 1 - debug: конфигурационные данные (при запуске системы), другая информация, необходимая для отладки, + сообщения уровня Info;
- 2 - info (значение по умолчанию): базовая информация (сообщения о запуске, работе, выключении системы) + сообщения уровня Warning;
- 3 - warning: системные предупреждения + сообщения уровня Error;
- 4 - error: все ошибки, возникающие в процессе работы, в том числе ошибки уровня Fatal;
- 5 - fatal: критические ошибки, приводящие к сбоям системы.

6.3. Формат записей

Все логи представлены в формате JSON. Каждое сообщение лога представлено в формате отдельной JSON-структуры с исчерпывающим набором полей.

Логи имеют следующую структуру:

```
"app":"adec_scrambler","app_ver":"VERSION","level":"info","module":"COMPONENT","msg":"MSG","time":"2022-03-10T13:12:39.119Z"
```

Параметр	Описание

app	Наименование системы.
app_ver	Последняя актуальная версия ADECScrambler.
level	Уровень логирования (см. Режимы ведения логов).
module	Логический компонент системы ADECScrambler. Возможные значения: provider, scrambler, reshuffler, transmitter
msg	Комментарий к логу. Значение может отличаться в зависимости от компонента (значения module). **Особенности**: • Только для сочетания (module: "provider" + level: "trace"): *NEW PID: number_pid*. Например, "NEW PID: 123" означает, что компонент Provider нашел PID под номером 123 и начал читать данные по этому PID'у. • Для всех компонентов и level: "trace": *(transmitter speed: 123 Kb/sec)*, *(provider speed: 123 Kb /sec)* и т.д. В сообщении показана скорость работы (битрейт) компонента. • Для всех компонентов: *(scrambler loop was started)*, *(provider loop was started)* и т.д. Данное сообщение появляется в самом начале, когда компонент создан. • Только для scrambler: *stopped*. Данное сообщение появляется, когда останавливаем контейнер. • *stopping the server*. Данное сообщение появляется, только когда останавливаем сервер.
time	Дата и время (вплоть до миллисекунд) формирования записи (фиксации лога).

Пример лога (остановка сервера):

```
"app":"adec_scrambler","app_ver":"1.0.12\n","level":"info","msg":"stopping the server","time":"2022-03-10T13:49:46.96Z"
```

Перейти к Содержанию...

7. Сбор метрик

7.1. Общее описание

ADECScrambler собирает статистику о своей работе. Показатели, по которым собирается статистика, задаются так называемыми "метриками".

Сбор метрик осуществляется средствами Prometheus, где собранная информация аккумулируется и отображается графически с помощью Grafana.

7.2. Типы метрик

Метрики Prometheus бывают следующих типов:

- Counter
- Gauge
- Histogram
- Summary

Counter - счетчик, значение которого может увеличиваться либо сбрасываться до нуля (только при перезапуске). Используется для представления количества обслуженных запросов, выполненных задач, ошибок и т.д. Counter не используется для отображения значений, которые могут уменьшиться (например, для количества запущенных в данный момент процессов); в этом случае вместо counter используется gauge.

Gauge - числовое значение, которое может произвольно увеличиваться или уменьшаться. Обычно используется для измерения значений (например, температура, текущее использование памяти), а также для "счетчиков", которые могут увеличиваться или уменьшаться (например, количество одновременных запросов).

Histogram - показывает наблюдения (например, длительность запросов или размер ответов) и подсчитывает их в настраиваемых сегментах. Histogram также предоставляет сумму всех наблюдаемых значений.

Совместно с базовой метрикой (в примерах ниже - <basename>) histogram показывает несколько временных интервалов:

- <basename>_bucket {le = "<верхняя включающая граница>"} - совокупные счетчики для сегментов наблюдения;
- <basename>_sum - общая сумма всех наблюдаемых значений;
- <basename>_count (идентично <basename> _bucket {le = "+ Inf"} выше) - количество наблюдаемых событий.

Summary - подобно histogram, аккумулирует наблюдения (как правило, длительность запросов и размер ответов). Хотя этот тип метрики предоставляет общее количество наблюдений и сумму всех наблюдаемых значений, но он также вычисляет настраиваемые квантили в скользящем временном окне.

Совместно с базовой метрикой (в примерах ниже - <basename>) summary показывает несколько временных интервалов:

- <basename> {quantile = "<φ>"} - потоковая передача φ-квантилей (0 ≤ φ ≤ 1) наблюдаемых событий;
- <basename>_sum - общая сумма всех наблюдаемых значений;
- <basename>_count - количество наблюдаемых событий.

Описанные выше типы метрик предоставляют клиентские библиотеки Prometheus. При этом сервер Prometheus пока еще не использует информацию о типе и объединяет все данные в не типизированные временные ряды.

Более подробную информацию можно найти здесь: https://prometheus.io/docs/concepts/metric_types/

7.3. Prometheus метрики

7.3.1. go_*

Метрика	Тип	Описание
go_gc_duration_seconds	summary	Сводная информация (summary) о продолжительности паузы в циклах "сборки мусора" (освобождение памяти от ненужных, устаревших, неверных или разрушенных (например, в результате сбоя) данных). **Примечание**. У метрики с типом summary есть дополнительный параметр quantile (квантиль), с его помощью метрика одного типа разбивается на квантили: • go_gc_duration_seconds{quantile="0"} • go_gc_duration_seconds{quantile="0.25"} • go_gc_duration_seconds{quantile="0.5"} • go_gc_duration_seconds{quantile="0.75"} • go_gc_duration_seconds{quantile="1"}
go_goroutines	gauge	Количество существующих в данный момент горутин (гоурутина - средство для реализации параллелизма в языке *Go*).
go_info	gauge	Информация об окружении *Go*.
go_memstats_alloc_bytes	gauge	Количество байт, выделенных и все еще используемых.
go_memstats_alloc_bytes_total	counter	Общее количество выделенных байт, даже если они освобождены.
go_memstats_buck_hash_sys_bytes	gauge	Количество байт, используемых хеш-таблицей сегмента профилирования.
go_memstats_frees_total	counter	Общее количество свободных (байт).
go_memstats_gc_cpu_fraction	gauge	Доля доступного процессорного (CPU) времени этой программы, использованная "сборщиком мусора" (GC) с момента запуска программы.
go_memstats_gc_sys_bytes	gauge	Количество байт, используемых для метаданных системы "сборки мусора" (системы очистки памяти от ненужных данных).

go_memstats_heap_alloc_bytes	gauge	Количество байт динамической памяти (heap bytes), выделенных и все еще используемых.
go_memstats_heap_idle_bytes	gauge	Количество байт динамической памяти (heap bytes), ожидающих использования.
go_memstats_heap_inuse_bytes	gauge	Количество используемых байт динамической памяти (heap bytes).
go_memstats_heap_objects	gauge	Количество выделенных объектов.
go_memstats_heap_released_bytes	gauge	Количество байт динамической памяти (heap bytes), выпущенных для ОС.
go_memstats_heap_sys_bytes	gauge	Количество байт динамической памяти (heap bytes), полученных от системы.
go_memstats_last_gc_time_seconds	gauge	Количество секунд с последней "сборки мусора".
go_memstats_lookups_total	counter	Общее количество поисков указателя (pointer lookups).
go_memstats_mallocs_total	counter	Общее количество выделений памяти (mallocs).
go_memstats_mcache_inuse_bytes	gauge	Количество байт, используемых структурами *mcache*.
go_memstats_mcache_sys_bytes	gauge	Количество байт, используемых для структур *mcache*, полученных из *system*.
go_memstats_mspan_inuse_bytes	gauge	Количество байт, используемых структурами *mspan*.
go_memstats_mspan_sys_bytes	gauge	Количество байт, используемых для структур *mspan*, полученных из *system*.
go_memstats_next_gc_bytes	gauge	Количество байт кучи (heap bytes), когда будет произведена следующая "сборка мусора".
go_memstats_other_sys_bytes	gauge	Количество байт, используемых для других системных распределений.
go_memstats_stack_inuse_bytes	gauge	Количество байт, используемых распределителем стека (stack allocator).
go_memstats_stack_sys_bytes	gauge	Количество байт, полученных от системы для распределителя стека (stack allocator).
go_memstats_sys_bytes	gauge	Количество байт, полученных от системы.
go_threads	gauge	Количество созданных потоков ОС.

7.3.2. process_*

Метрика	Тип	Описание

process_cpu_seconds_total	counter	Общее время (в секундах), затраченное пользователем и системным процессором.
process_resident_memory_bytes	gauge	Размер резидентной памяти (в байтах).
process_start_time_seconds	gauge	Время начала процесса с эпохи unix (в секундах).
process_virtual_memory_bytes	gauge	Размер виртуальной памяти (в байтах).
process_virtual_memory_max_bytes	gauge	Максимальный объем доступной виртуальной памяти (в байтах).

7.3.3. adec_scrambler_*

Метрика	Тип	Описание
adec_scrambler_go_crypt_operation_full		Число принятых байт на шифрование. Учитываются только байты из payload пакета.
adec_scrambler_go_crypt_operation_success		Число успешно зашифрованных байт. Учитываются только байты из payload пакета, т.е. те, которые именно шифровались.
adec_scrambler_go_speed_meter_provider		Число байт, прошедших через provider.
adec_scrambler_go_speed_meter_scrambler		Число байт, прошедших через scrambler.
adec_scrambler_go_speed_meter_transmitter		Число байт, прошедших через transmitter.

Перейти к Содержанию...