АО «Концерн ГРАНИТ»

Россия, 119019, г. Москва, ул. Гоголевский бульвар, д. 31, стр. 2, эт. 2, пом.1
т. +7 495 642 97 42, ф. +7 499 558 15 29
office@granit-concern.ru, granit-concern.ru

QUANTUM SECURE STORAGE

Руководство системного программиста

Листов 31

2023

Акционерное общество «Концерн ГРАНИТ»
ОКПО 78089277 к/с 30101810400000000225, БИК 044525225
, ОГРН 1055011347093, ИНН/КПП 5003056699/770401001
р/с 40702810738000014569 в ПАО «Сбербанк России»,

АННОТАЦИЯ

Настоящий документ содержит сведения по установке и настройке, командах административного интерфейса «Quantum Secure Storage» (далее QSS, Программа), предназначенной для криптографической защиты конфиденциальности и целостности информации, в том числе для защиты персональных данных. Кроме того, в документ содержит общие данные по QSS: описание функциональности и структуры.

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

1.1. Назначение

Программа предназначена для криптографической защиты конфиденциальности и целостности информации, в том числе для защиты персональных данных, и представляет собой программный продукт на языке Rust со встроенной библиотекой шифрования, консольными и программными интерфейсами. Криптографическая защита должна соответствовать требованиям ГОСТ: ГОСТ Р 34.13-2015, ГОСТ 34.13-2018, ГОСТ Р 34.12-2015, ГОСТ 34.12-2018, ГОСТ Р 34.11-2012, ГОСТ 34.11-2018 и стандартам: Р 1323565.1.026–2019, Р 50.1.111-2016, Р 50.1.113-2016. Система предназначена для защиты конфиденциальности и целостности информации, не содержащей сведений, составляющих государственную тайну.

1.2. Функции

Программа обеспечивает выполнение следующих функций:

1) аутентификацию при установлении административного соединения по паролю и дополнительной случайной информации (с опциональным её хранением на внешнем носителе) с использованием алгоритма выработки ключа из пароля по алгоритму «PBKDF2» в соответствии с «Р 50.1.111-2016»;
2) диверсификацию ключей с использованием алгоритма KDF_GOSTR3411_2012_256 в соответствии «Р 50.1.113-2016»;
3) аутентифицированное шифрование/расшифрование данных (с использованием разделяемой памяти/ через сокеты) в соответствии с «Р 1323565.1.026—2019» (алгоритмом «Кузнечик» в соответствии «ГОСТ 34.12—2018»);
4) формирование и проверку электронной подписи в соответствии с «ГОСТ 34.10-2018» и параметрами эллиптических кривых, определёнными в «Р 1323565.1.024-2019»;

5) формирование общих ключей по алгоритму «VKO» в соответствии с «Р 50.1.113-2016»;
6) вычисление хеш-суммы от блока данных в соответствии с «ГОСТ 34.11-2018»;
7) управление ключевой информацией;
8) хранение ключей в зашифрованном и имитозащищенном виде в долговременном хранилище (диске);
9) стирание из оперативной памяти ключевой информации после окончания её использования путем перезаписи псевдослучайной последовательностью;
10) защиту ключевой информации в оперативной памяти путем ее маскирования;
11) контроль жизненного цикла ключа: запрет техническими средствами на использование ключа для шифрования и формирования цифровой подписи после истечения времени его жизни (не может составлять более 15 месяцев) и заблаговременное уведомление администраторов о необходимости смены ключа.
12) управление администраторами.

Функциональные требования по шифрованию реализованы в соответствии с ГОСТ: ГОСТ Р 34.13-2015, ГОСТ 34.13-2018, ГОСТ Р 34.12-2015, ГОСТ 34.12-2018, ГОСТ Р 34.11-2012, ГОСТ 34.11-2018, и стандартами: Р 1323565.1.026–2019, Р 50.1.111-2016, Р 50.1.113-2016.

Реализовано использование алгоритма Multilinear Galous Mode, описанного в Р 1323565.1.026–2019, в комбинации с шифром «Кузнечик», описанным в ГОСТ Р 34.12-2015 и ГОСТ 34.12-2018. При шифровании данных используется вектор инициализации, сгенерированный ГПСЧ на основе блочного шифра «Кузнечик» в режиме гаммирования.

1.3. Условия выполнения

Система функционирует на ЭВМ с характеристиками, не ниже следующих:

– процессоры архитектуры (только для 64 битных CPU): x86-64 с тактовой частотой не менее 2 ГГц;

- оперативная память: не менее 4 Гб оперативной памяти;
- жёсткий диск: не менее 1 Гб.

Система функционирует в среде на базе следующих ОС:

- CentOS 7 и 8;
- РЕД ОС;
- ROSA Enterprise Linux Server (RELS);
- РОСА «Кобальт»;
- Альт 8 СП;
- Альт Сервер 9;
- Fedora 33, 34 и 35;
- Debian 9 и 10;
- Astra Linux Special Edition «Смоленск» 1.6, 1.7;
- Ubuntu 18.04 LTS и 22.04 LTS;
- openSUSE 15.4.

2. СТРУКТУРА

Программный комплекс представляет собой программный продукт, реализованный на языке Rust, со встроенной библиотекой, реализующей криптографические алгоритмы, консольным и программными интерфейсами, опционально использующим в качестве внешних библиотек решения поставщиков ФДСЧ.

Структура Программы и используемых внешних решений включает в себя такие элементы, как:

1. qss-server (файл qss) – элемент выступает в роли сервера, отвечающего на команды, посылаемые пользователем;
2. qss-client – элемент представляет собой пользовательский консольный интерфейс для работы с серверной частью;
3. libqss.so – пользовательская библиотека для работы с qss-server через API;
4. qss-admin – элемент представляет собой административный консольный интерфейс для работы с серверной частью;
5. libgostcrypto.so – динамическая библиотека, реализующая криптографические алгоритмы;
6. config.yml – файл с настройками;
7. storage.bin – файл с реестром пользователей, зашифрованными рабочими ключами, счетчиками попыток входа;
8. bin_integrity.streebog512 — файл с эталонными значениями хеш-кодов библиотек и исполняемых файлов являющихся частью СКЗИ;
9. журналы регистрации событий;
10. (опционально) внешние библиотеки:
    a. libsobol.so – используется для получения случайных последовательностей, сгенерированных на ПАК «Соболь»;
    b. libtmdrv.so – используется для получения случайных последовательностей, сгенерированных на СЗИ НСД «Аккорд-АМДЗ»;

с. иные библиотеки для получения случайных последовательностей.

Структурная схема зависимостей представлена на рисунке 1.

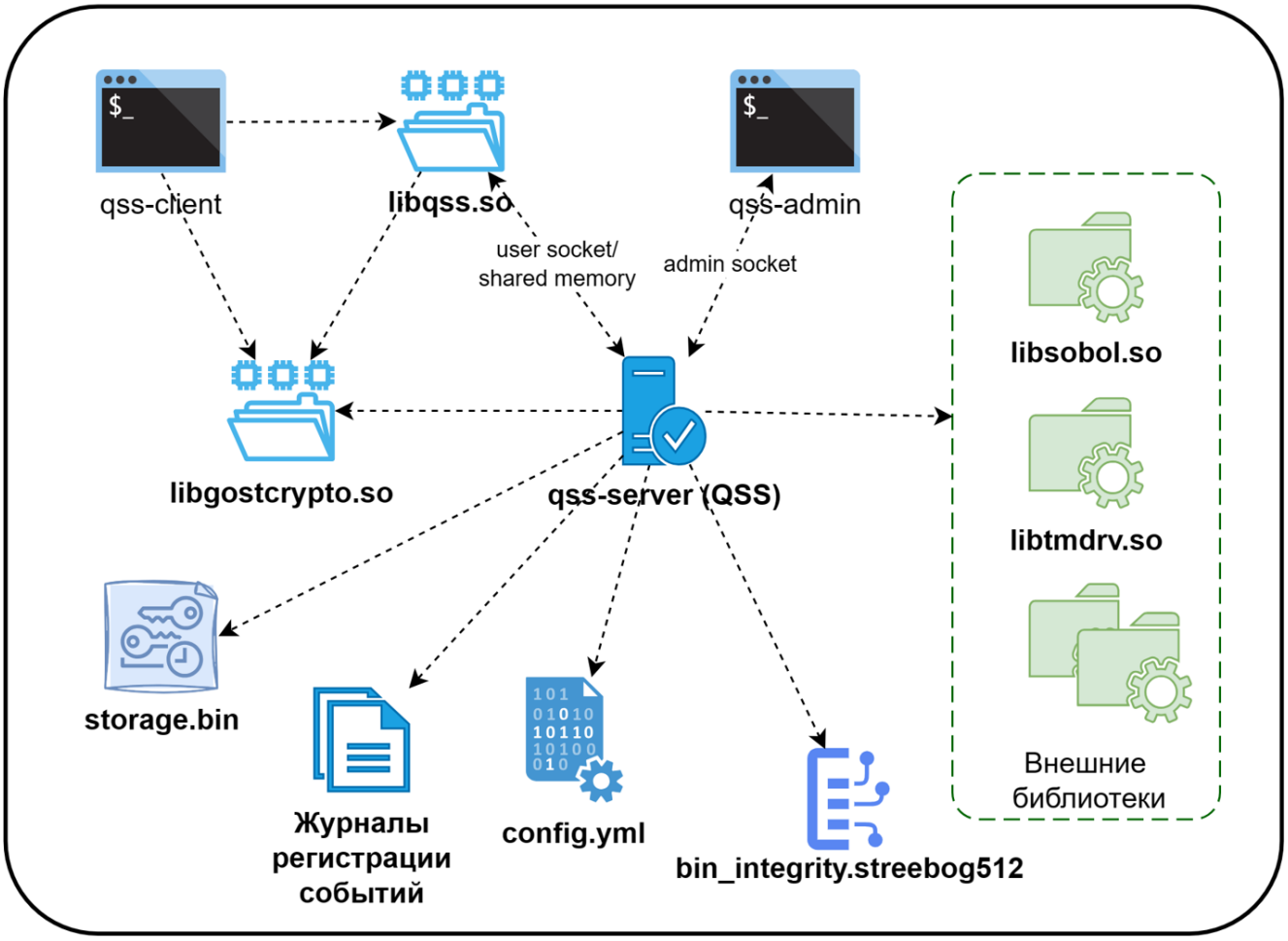

Рисунок 1 - Зависимости QSS

Исходя из рисунка 1 видно, что qss-server (файл qss) и интерфейсы пользователя и администратора (библиотека libqss.so и консольный интерфейс qss-admin) взаимодействуют между собой по клиент-серверной модели посредством двух UNIX сокетов: пользовательского и административного. Кроме того, при шифровании и расшифровании имеется возможность использовать разделяемую память. Настройки для qss-server хранятся в файле config.yml. В качестве элемента, реализующего криптографические алгоритмы, используется библиотека libgostcrypto.so. Кроме вышеперечисленных элементов рисунка qss-server использует storage.bin для хранения рабочей информации о пользователях и рабочих ключах и (опционально) внешние библиотеки (например, libsobol.so, libtmdrv.so) для доступа к

физическому датчику случайных чисел (далее ФДСЧ), поставляемые вместе со сторонними решениями, такими как ПАК «Соболь», СЗИ НСД «Аккорд-АМДЗ».

Работа пользователя с Программным комплексом предусмотрена через CLI администратора для административных задач и через API для прямого обращения приложений к криптографическим функциям.

СКЗИ представляет собой готовый продукт. Имеется возможность использовать СКЗИ в качестве встраиваемого решения путем обращения к функциям программного комплекса через библиотеки libqss.so и libgostcrypto.so, минуя qss-client.

Для обеспечения доверенной загрузки совместно с СКЗИ QSS необходимо использовать механизм доверенной загрузки, имеющий сертификат соответствия ФСБ России по классу не ниже 2Б.

Для генерации случайных последовательностей СКЗИ QSS использует ФДСЧ, имеющий сертификат соответствия ФСБ России по классу не ниже 2Б (ФДСЧ из состава механизмов доверенной загрузки, имеющих сертификаты соответствия ФСБ России по классу 2Б и выше).

3. ПРОВЕРКА

При приёмке СКЗИ в случае получения дистрибутива конечный пользователь должен проверить:

- целостность упаковки;
- комплектность (наличие дистрибутива, электронного варианта эксплуатационной документации и формуляра);
- идентичность учётных номеров СКЗИ на дистрибутиве и в формуляре;
- целостность полученного дистрибутива путём вычисления контрольных сумм файлов дистрибутива с использованием утилиты ФИКС и сравнения вычисленных контрольных сумм с зафиксированными в формуляре.

При установке СКЗИ не должно быть сообщений об ошибках (исключения описаны в разделе 9 «Сообщения системному администратору»).

Сообщения об ошибках АМДЗ не относятся к корректности работы СКЗИ и должны решаться с технической поддержкой компании-разработчика АМДЗ.

4. УСТАНОВКА И НАСТРОЙКА

При изменении конфигурации в части расположения в файловой системе файлов, используемых СКЗИ QSS, необходимо обеспечить те же права доступа к ним, что и права по умолчанию.

4.1. Установка QSS

Для установки QSS необходимо выполнить действия в следующей последовательности:

Для установки QSS необходимо выполнить последовательность шагов.

1. Включить питание ЭВМ, оборудованной МДЗ с установленной ОС из списка в разделе 1.3 настоящего документа.
2. Получить необходимые пакеты для установки в зависимости от используемой ОС:
   a. для Alt Linux/CentOS /Fedora/ openSUSE/ РЕД ОС/ RELS/ РОСА «Кобальт» - пакеты .rpm;
   b. для Astra Linux/Debian - пакеты .deb.
3. Выполнить установку (под правами администратора):
   a. для установки .rpm пакетов воспользуйтесь командами:

      для всех ОС, кроме Alt Linux

      sudo yum install ./libgostcrypto-0.3.1-1.x86_64.rpm -y

      sudo yum install ./qss-0.1.0-1.el7.x86_64.rpm -y

      для Alt Linux

      sudo apt-get install ./libgostcrypto-0.3.1-1.x86_64.rpm -y

      sudo apt-get install ./qss-0.1.0-1.el7.x86_64.rpm -y

   b. для установки .deb пакетов воспользуйтесь командами:

      sudo apt-get install ./libgostcrypto_0.3.1_amd64.deb -y

      sudo apt-get install ./qss_0.1.0_amd64.deb -y
4. Добавить пользователей ОС, осуществляющих администрирование QSS, в группу qss-admin, используя следующую команду (замените username на имя пользователя): *sudo usermod -a -G qss-admin username*

5. Добавить пользователей ОС, от которых будут осуществляться клиентские действия с QSS, в группу qss-client, используя следующую команду (замените username на имя пользователя): *sudo usermod -a -G qss-client username*
6. Проинспектировать файл настроек (см. раздел 4.2), располагающийся по пути */etc/qss/config.yml*, сменить опцию *hw_rng* с *Os* на используемый ФДСЧ (может потребоваться установка библиотек от производителя), внести иные изменения при необходимости (см. раздел 4.2 для описания настроек QSS)
7. Добавить в файл с контрольными хеш-суммами бинарных файлов хеш-сумму и путь к библиотеке используемой для доступа к ФДСЧ. При использовании ПАК «Соболь» и «Аккорд» используются библиотеки *libsobol.so* и *libtmdrv.so* соответственно. Хеш-сумма может быть вычислена используя команду (замените */path/to/lib* на путь к библиотеке): *qss-client streebog512 --skip-bin-integrity /path/to/lib*. Обратите внимание, что при внесении полученного значения в файл хеш-сумма и путь должны быть разделены символом табуляции, а не пробелами.
8. Запустить QSS как SystemD сервис: *sudo systemctl start qss.service*
9. Произвести инициализацию хранилища QSS из-под пользователя, входящего в группу qss-admin (замените admin_username на имя пользователя): *qss-admin --init admin_username*
10. Выйти из административного интерфейса используя команду exit, либо используя комбинацию клавиш *Ctrl+C* или *Ctrl+D*
11. Проверить авторизацию в административном интерфейсе (замените *admin_username* на имя администратора): *qss-admin admin_username*
12. Создать рабочие ключи в административном интерфейсе используя команду (типы ключей можно получить используя встроенную справку, либо используя автодополнение по нажатию *Tab*): *key create <key_type> <key_label>*
13. Разблокировать рабочие ключи в административном интерфейсе используя команду: *key unlock <key_label>*
14. Проверить статус рабочих ключей в административном интерфейсе используя команду: *key list*
15. Выйти из административного интерфейса.

4.2. Описание настроек QSS

admin_log_level

Уровень логирования для административного сервиса

Возможные значения: trace, debug, info, warn, error. Каждый из указанных значений параметра имеет свой набор логируемых событий, плюс включает все события вложенного уровня (error входит в уровень warn, warn в info, info в debug, debug в trace). Таким образом уровень trace отображает события всех уровней.

Info: уровень по умолчанию. Отображает события связанные с конфигурационным файлом (проверка при запуске), все принятые команды, все отправляемые ответы, показывает сообщение об успешной инициализации/авторизации, тип авторизации, а также события, связанные с остановкой и завершением работы QSS, запуском внутренних потоков, установкой соединений и разъединением соединений. На этом уровне отображаются также все события уровня Error и Warn.

Warn: уровень отображения событий-предупреждений. Отображает предупреждения, если не удалось авторизовать администратора с указанием причины (не существует с таким именем или превышено количество неудачных попыток и т.п.), предупреждения при попытке установить ещё одно административное соединение (в QSS разрешено только одно соединение в конкретный момент времени), предупреждения при работе с разделяемой памятью, предупреждения при истечении времени параметра idle_admin_session_timeout. На этом уровне отображаются также все события уровня Error.

Error: уровень отображения событий об ошибках. На данном уровне выводятся только сообщения о критических событиях: невозможно запустить сервер, не удаётся получить случайное значение, критическая ошибка в каком-либо потоке, ошибки при зашифровании, ошибки при попытках использования ключа т.к. ключ заблокирован, ошибки, связанные с тем, что указанные ключи уже существуют, ошибки при выполнении команд по копированию\перемещении\удалению логов, ошибки при приеме команд, невозможно заменить пользователя при его отсутствии, невозможно отправить ответное сообщение, невозможно удалить файл-сокет (при завершении

работы), ошибки при работе с unix socket, завершение QSS с любой ошибкой, а также все ответы QSS, в которых сообщения о проблемах фиксируются (например, неправильный пароль).

Debug: уровень отображения событий для отладки работы QSS. На данном уровне отображаются сообщения при выполнении зашифрования с ключом, события получения сообщений с длиной последующей команды; отображаются события, указывающие частичное прочтение QSS сообщений; события, связанные с транспортировкой данных, указывается информация о сообщение с указанием его длины, указывается информация об отправке команд с указанием их длины. На этом уровне отображаются также все события уровня Error, Warn, Info.

Trace: уровень отображения событий для отладки с детализацией по этапам обработки сообщений. Отображает события по сборке сообщений (для больших сообщений). На этом уровне отображаются также все события уровня Error, Warn, Info, Debug.

client_log_level

Уровень журналирования для криптографического сервиса.

Работа параметра аналогична admin_log_level: совпадает «вложенность» значений параметра, исключение – отсутствие уровня trace.

Debug: отображает события о том, что сообщение прочитано только частично. На этом уровне отображаются также все события уровня Error, Warn, Info.

Info: отображает события о принятии команда и отправке ответа; события запуска потоков, занимающихся транспортом, логикой, проверкой целостности; события завершения работы потока(ов), сообщения о том, что удалён файл, представляющий собой сокет; события принятия соединений и закрытия соединений. На этом уровне отображаются также все события уровня Error и Warn.

Warn: уровень отображения событий-предупреждений. Отдельных событий на этом уровне не предусмотрено. На этом уровне отображаются также все события уровня Error.

Error: уровень отображения событий об ошибках. На данном уровне выводятся только сообщения о критических событиях: обнаружено нарушение проверки

целостности шифра, ошибки при обработке команд, ошибки при транспортировке (unix socket), ошибки отправки ответа: соединение будет закрыто, ошибки при удалении unix socket файлов (при закрытии), не удалось десериализовать команду.

integrity_log_level

Уровень журналирования для сервиса проверки целостности

Возможные значения: trace, debug, info, warn, error.

integrity_data

Путь к файлу с хеш-кодами бинарных файлов

hw_rng

Физический датчик случайных чисел

Возможные значения: Os, Sobol, Accord, Wrapper. При работе СКЗИ должно быть установлено одно из трех значений данного параметра: Sobol, Accord, Wrapper.

При использовании Wrapper должны выполняться следующие условия:

- наличие у датчика случайных чисел заключения или сертификата соответствия ФСБ России;
- ФДСЧ должен поддерживать ОС, на которой установлен ПК QSS;
- поддержка сертифицированной библиотекой датчика случайных чисел интерфейса СКЗИ QSS;
- отсутствие необходимости проведения оценки влияния на датчик случайных со стороны СКЗИ QSS.

При выборе опции Wrapper необходимо указать путь к библиотеке, имя функции генерации случайной последовательности и (опционально) имя функции инициализации. Пример:

```
hw_rng:
  !Wrapper
    lib_path: "/usr/lib/libfoobar.so"
    init_fn: "foobar_init"
    getrandom_fn: "foobar_getrandom"
```

Сигнатуры функции инициализации и генерации должны соответствовать следующему хедеру (имена функций могут быть иными):

uint32_t foobar_init();

uint32_t foobar_getrandom(uint8_t *p, size_t len);

Код возврата равный нулю интерпретируется как успешный результат, остальные значения интерпретируются как код ошибки. Функция инициализации, если она указана в конфигурации, вызывается единожды при старте QSS до вызова функции генерации. Функция генерации при возврате нуля должна заполнить буфер длиной len соответствующий указателю p случайной последовательностью полученной от ФДСЧ.

log_path

Путь к папке для сохранения журнала регистрации событий

log_rotate_size

Размер файла журнала регистрации событий при котором производится его ротация

При достижении данного размера файлом текущего журнала, происходит сжатие текущего файла и журналирование продолжается в новом файле.

log_files_keep

Количество файлов журнала регистрации событий, которые нужно сохранять.

При использовании ПК QSS данный параметр должен быть пустым. В случае пропуска данной опции, либо при указании пустого значения, сохраняются все файлы журнала.

client_address

Адрес клиентского сокета QSS

admin_address

Адрес административного сокета QSS

client_group

Группа для пользовательского сокета QSS

В случае пропуска данной опции, либо при указании пустого значения, группа для пользовательского сокета не изменяется и остаётся равной пользователю под которым запущен QSS.

admin_group

Группа для административного сокета QSS

В случае пропуска данной опции, либо при указании пустого значения, группа для административного сокета не изменяется и остаётся равной пользователю под которым запущен QSS.

storage_path

Путь к хранилищу QSS.

client_stack_size

Размер стека клиентских тредов.

Рекомендуется использовать значение равное степени двойки. Допустимые единицы измерений: B, KiB, MiB.

При использовании ПК QSS рекомендованное значение 16 KiB.

admin_stack_size

Размер стека тредов администратора.

Рекомендуется использовать значение равное степени двойки. Допустимые единицы измерений: B, KiB, MiB.

При использовании ПК QSS рекомендованное значение 16 KiB.

mlockall

Возможные значения: true, false

При использовании ПК QSS данный параметр должен быть true. При установке в true запрещается вытеснение оперативной памяти процесса QSS в swap посредством вызова mlockall(MCL_CURRENT | MCL_FUTURE). Данный вызов требует либо привилегии CAP_IPC_LOCK, либо установки значения RLIMIT_MEMLOCK.

working_key_expiration_reminder_period

Период, начиная с которого должно быть произведено напоминание о скором истечении времени жизни рабочего ключа

admin_password_expiration_reminder_period

Период, начиная с которого должно быть произведено напоминание о скором истечении срока действия пароля администратора

integrity_check_period

Периодичность проверки целостности ключей и криптографических алгоритмов. При использовании ПК QSS данный параметр должен принадлежать интервалу [0, 5] min.

idle_admin_session_timeout

Период неактивности администратора, по истечении которого соединение с администратором будет закрыто.

В случае пропуска данной опции, либо при указании пустого значения, таймаут не устанавливается (иначе говоря, становится равен бесконечности).

socket_read_timeout

Таймаут чтения из сокета (периодичность проверки флага останова)

В случае пропуска данной опции, либо при указании пустого значения, таймаут не устанавливается (иначе говоря, становится равен бесконечности).

socket_write_timeout

Таймаут записи в сокет

В случае пропуска данной опции, либо при указании пустого значения, таймаут не устанавливается (иначе говоря, становится равен бесконечности).

num_tries_reg_control

Количество попыток прохождения регламентного контроля для датчиков случайных чисел.

num_tries_hw_rng

Количество попыток прохождения статистического контроля для ФДСЧ

num_tries_prng_ctr

Количество попыток прохождения статистического контроля для ГПСЧ на основе гаммирования

num_tries_prng_xorshift

Количество попыток прохождения статистического контроля для ГПСЧ на основе регистра сдвига

5. ПЕРЕЧЕНЬ КОМАНД АДМИНИСТРАТИВНОГО ИНТЕРФЕЙСА (QSS-ADMIN)

Команды:

- *exit* Выход
- *help* Помощь
- *key* Управление ключами
- *log* Управление логами
- *shmem* Управление разделяемой памятью
- *storage* Управление хранилищем
- *admin* Управление администраторами
- *check* Регламентный контроль

a. key Управление ключами

   key <SUBCOMMAND>

   Подкоманды (SUBCOMMANDS):

   - *delete* Удалить ключ QSS
   - *list* Вывести список ключей QSS
   - *lock* Заблокировать имеющиеся ключи QSS
   - *create* Создать новый ключ QSS
   - *unlock* Разблокировать имеющиеся ключи QSS
   - *help* Помощь

b. log Управление логами:

   log <SUBCOMMAND>

   Подкоманды (SUBCOMMANDS):

   - *list* Получить список файлов журнала
   - *rotate* Произвести ротацию текущего файла журнала
   - *copy* Скопировать ротированные файлы журнала по указанному пути
   - *move* Переместить ротированные файлы журнала по указанному пути
   - *delete* Удалить ротированные файлы журнала
   - *help* Помощь

c. admin Управление администраторами

admin <SUBCOMMAND>

Подкоманды (SUBCOMMANDS):

- *add* Добавить нового администратора QSS
- *delete* Удалить администратора QSS
- *edit* Изменить администратора QSS
- *list* Показать существующих администраторов QSS
- *help* Помощь

d. shmem Управление разделяемой памятью

shmem <SUBCOMMAND>

Подкоманды (SUBCOMMANDS):

- *list* Показать существующие именованные фрагменты разделяемой памяти
- *unlink* Отсоединить именованные фрагменты разделяемой памяти
- *help* Помощь

e. check Регламентный контроль

- *all* Запустить все проверки
- *crypto* Проверить работоспособность криптографических алгоритмов
- *integrity* Проверить целостность бинарных файлов
- *keys* Проверить целостность рабочих ключей
- *rng* Проверить датчики случайных чисел
- *version* Получить версию QSS и протоколов
- *help* Помощь

f. storage Управление хранилищем

storage-key <SUBCOMMAND>

Подкоманды (SUBCOMMANDS):

- *change-key* Изменить ключ хранения QSS
- *help* Помощь

6. ПЕРЕЧЕНЬ КОМАНД ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА (QSS-CLIENT)

Список команд:

- *key-kind* — Получить тип ключа
- *mgm128-encrypt* — Шифрование данных используя MGM с размером блока 128 бит
- *mgm128-decrypt* — Расшифрование данных используя MGM с размером блока 128 бит
- *mgm128-reencrypt* — Перешифрование данных используя MGM с размером блока 128 бит
- *sign256* — Формирование подписи (256 бит)
- *sign512* — Формирование подписи (512 бит)
- *streebog256* — Хеширование Стрибог-256
- *streebog512* — Хеширование Стрибог-512
- *verify-sig* — Верификация подписи
- *get-verif-key* — Получение ключа верификации
- *vko256-encrypt* — Выработать 256 битный общий ключ используя 256-битную кривую и зашифровать им данные
- *vko256-decrypt* — Выработать 256 битный общий ключ используя 256-битную кривую и расшифровать им данные
- *vko512-encrypt* — Выработать 256 битный общий ключ используя 512-битную кривую и зашифровать им данные
- *vko512-decrypt* — Выработать 256 битный общий ключ используя 512-битную кривую и расшифровать им данные
- *help* — Помощь

Список флагов каждой из команд можно узнать, используя команду:

qss-client help <имя команды>

7. СПИСОК ФУНКЦИЙ LIBQSS

- *qss_connect* — Установить соединение с QSS
- *qss_shrink_buffer* — Уменьшить размер внутреннего буфера в дескрипторе соединения QSS до заданного значения
- *qss_disconnect* — Разорвать соединение с QSS
- *qss_key_create* — Создать случайный сессионный ключ
- *qss_key_delete* — Удалить сессионный ключ
- *qss_key_kind* — Получить тип ключа
- *qss_shmem_open* — Подключить разделяемую память
- *qss_shmem_close* — Отключить разделяемую память
- *mgm128_decrypt_socket* — Расшифровать данные по сокету используя MGM с размером блока 128 бит
- *mgm128_encrypt_socket* — Зашифровать данные по сокету используя MGM с размером блока 128 бит
- *mgm128_reencrypt_socket* — Перешифровать данные по сокету используя MGM с размером блока 128 бит
- *mgm128_decrypt_shmem* — Расшифровать данные через разделяемую память используя MGM с размером блока 128 бит
- *mgm128_encrypt_shmem* — Зашифровать данные через разделяемую память используя MGM с размером блока 128 бит
- *mgm128_reencrypt_shmem* — Перешифровать данные через разделяемую память используя MGM с размером блока 128 бит
- *qss_sign_hash256* — Сформировать подпись для хеш-кода сообщения (256 бит)
- *qss_sign_hash512* — Сформировать подпись для хеш-кода сообщения (512 бит)
- *qss_sign_message256* — Сформировать подпись для сообщения (256 бит)
- *qss_sign_message512* — Сформировать подпись для сообщения (512 бит)
- *qss_get_verif_key256* — Получить ключ верификации (256 бит)
- *qss_get_verif_key512* — Получить ключ верификации (512 бит)
- *qss_vko256* — Выработать 256 битный общий ключ (256 бит)
- *qss_vko512* — Выработать 512 битный общий ключ (512 бит)

– *qss_vko512_256* — Выработать 256 битный общий ключ (512 бит)

Более подробная документация, списков аргументов и возвращаемые ошибки могут быть найдены в файле libqss.h, который входит в пакеты QSS.

8. СПИСОК ФУНКЦИЙ LIBGOSTCRYPTO

- *streebog256_new* — Создать состояние функции хэширования Стрибог-256
- *streebog256_digest* — Хэшировать сообщение алгоритмом Стрибог-256
- *streebog256_update* — Обновить состояние функции хэширования Стрибог-256
- *streebog256_finalize* — Финализировать состояние функции хэширования Стрибог-256
- *streebog256_finalize_reset* — Финализировать состояние функции хэширования Стрибог-256 и заменить его новым
- *streebog256_reset* — Сбросить состояние функции хэширования Стрибог-256
- *streebog256_drop* — Уничтожить состояние функции хэширования Стрибог-256
- *streebog512_new* — Создать состояние функции хэширования Стрибог-512
- *streebog512_digest* — Хэшировать сообщение алгоритмом Стрибог-512
- *streebog512_update* — Обновить состояние функции хэширования Стрибог-512
- *streebog512_finalize* — Финализировать состояние функции хэширования Стрибог-512
- *streebog512_finalize_reset* — Финализировать состояние функции хэширования Стрибог-512 и заменить его новым
- *streebog512_reset* — Сбросить состояние функции хэширования Стрибог-512
- *streebog512_drop* — Уничтожить состояние функции хэширования Стрибог-512
- *kuznyechik_enc_new* — Создать шифрующий контекст блочного шифра «Кузнечик»
- *kuznyechik_enc_drop* — Уничтожить шифрующий контекст блочного шифра «Кузнечик»

- *kuznyechik_mgm_encrypt* — Зашифровать сообщение используя блочный шифр «Кузнечик» в режиме MGM
- *kuznyechik_mgm_decrypt* — Расшифровать сообщение используя блочный шифр «Кузнечик» в режиме MGM
- *kuznyechik_mgm_reencrypt* — Перешифровать сообщение используя блочный шифр «Кузнечик» в режиме MGM
- *pbkdf2_hmac_streebog512* — Выработать ключ на основе пароля согласно Р 50.1.111-2016
- kdf*_gostr3411_2012_256* — Алгоритм диверсификации KDF_GOSTR3411_2012_256 определённый в Р 50.1.113-2016
- *ec_2012_256a_vko256*,
  ec_2012_256b_vko256,
  ec_2012_256c_vko256,
  ec_2012_256d_vko256 — Выработать общий 256-битный ключ используя соответствующую 256-битную кривую
- ec_2012_256a_serialize_verification_key,
  ec_2012_256b_serialize_verification_key,
  ec_2012_256c_serialize_verification_key,
  ec_2012_256d_serialize_verification_key — Сериализовать ключ верификации 256-битной кривой
- ec_2012_256a_deserialize_verification_key,
  ec_2012_256b_deserialize_verification_key,
  ec_2012_256c_deserialize_verification_key,
  ec_2012_256d_deserialize_verification_key — Десериализовать ключ верификации 256-битной кривой
- ec_2012_256a_serialize_signature_key,
  ec_2012_256b_serialize_signature_key,
  ec_2012_256c_serialize_signature_key,
  ec_2012_256d_serialize_signature_key — Сериализовать ключ подписи 256-битной кривой

- ec_2012_256a_deserialize_signature_key,
  ec_2012_256b_deserialize_signature_key,
  ec_2012_256c_deserialize_signature_key,
  ec_2012_256d_deserialize_signature_key — Десериализовать ключ подписи 256-битной кривой
- ec_2012_256a_verify_hash_sig,
  ec_2012_256b_verify_hash_sig,
  ec_2012_256c_verify_hash_sig,
  ec_2012_256d_verify_hash_sig — Проверить подпись хеш-кода сформированную 256-битной кривой
- ec_2012_256a_verify_msg_sig,
  ec_2012_256b_verify_msg_sig,
  ec_2012_256c_verify_msg_sig,
  ec_2012_256d_verify_msg_sig — Проверить подпись сообщения сформированную 256-битной кривой
- ec_2012_512a_vko256,
  ec_2012_512b_vko256,
  ec_2012_512c_vko256 — Выработать общий 256-битный ключ используя соответствующую 512-битную кривую
- ec_2012_512a_vko512,
  ec_2012_512b_vko512,
  ec_2012_512c_vko512 — Выработать общий 512-битный ключ используя соответствующую 256-битную кривую
- ec_2012_512a_serialize_verification_key,
  ec_2012_512b_serialize_verification_key,
  ec_2012_512c_serialize_verification_key — Сериализовать ключ верификации 512-битной кривой
- ec_2012_512a_deserialize_verification_key,
  ec_2012_512b_deserialize_verification_key,

ec_2012_512c_deserialize_verification_key — Десериализовать ключ верификации 512-битной кривой

- ec_2012_512a_serialize_signature_key,
  ec_2012_512b_serialize_signature_key,
  ec_2012_512c_serialize_signature_key — Сериализовать ключ подписи 512-битной кривой
- ec_2012_512a_deserialize_signature_key,
  ec_2012_512b_deserialize_signature_key,
  ec_2012_512c_deserialize_signature_key — Десериализовать ключ подписи 512-битной кривой
- ec_2012_512a_verify_hash_sig,
  ec_2012_512b_verify_hash_sig,
  ec_2012_512c_verify_hash_sig — Проверить подпись хеш-кода сформированную 512-битной кривой
- ec_2012_512a_verify_msg_sig,
  ec_2012_512b_verify_msg_sig,
  ec_2012_512c_verify_msg_sig — Проверить подпись сообщения сформированную 512-битной кривой

Более подробная документация, списков аргументов и возвращаемые ошибки могут быть найдены в файле libgostcrypto.h, который входит в пакеты libgostcrypto.

9. СООБЩЕНИЯ СИСТЕМНОМУ АДМИНИСТРАТОРУ

QSS выдает сообщения пользователям в ответ на команды в командном или консольном интерфейсах. Отдельные сообщения системному администратору в Программе не предусмотрены (исключения, рассмотрены в двух параграфах ниже).

Если при инициализации QSS (шаг 8, раздел 4) получена ошибка «Не удаётся установить соединение с QSS: No such file or directory (os error 2)», следует проверить запущен ли процесс QSS (например, используя команду *ps -ef | grep qss*). Если процесс qss не запущен, то следует проинспектировать логи QSS на предмет ошибок, приведших к остановке QSS. Если ошибка вызвана ФДСЧ, следует перепроверить установку и настройку ПАК, согласно инструкциям производителя.

Если пользовательское ПО возвращает ошибку «Ключ не был найден», то следует проверить правильность идентификатора ключа и был ли ключ с данным идентификатором разблокирован (используя административную команду *key list*). В случае, если ключ был заблокирован, следует его разблокировать используя команду *key unlock <key_label>*. Обратите внимание, что при перезагрузке QSS (в т.ч. при перезагрузке всей системы) все ключи переходят в заблокированное состояние.

Дополнительно необходимо обращать внимания на сообщения от средств доверенной загрузки при загрузке ЭВМ (описание необходимости регулярных проверок содержится в формуляре).

ПЕРЕЧЕНЬ ПРИНЯТЫХ СОКРАЩЕНИЙ

Термин/ Сокращение, обозначение	**Расшифровка**
CPU	Центральный процессор (с англ. «Central processing unit»)
QSS	Quantum Secure Storage Программа, предназначенная для криптографической защиты конфиденциальности и целостности информации, в том числе для защиты персональных данных
Гб	Гигабайт, единица измерения количества информации
ГОСТ	Государственный стандарт
ГОСТ Р 34.11-2012	ГОСТ Р 34.11-2012 Информационная технология. Криптографическая защита информации. Функция хэширования
ГОСТ 34.11-2018	ГОСТ 34.11-2018 Информационная технология. Криптографическая защита информации. Функция хэширования.
ГОСТ Р 34.12-2015	ГОСТ Р 34.12-2015 Информационная технология. Криптографическая защита информации. Блочные шифры.
ГОСТ 34.12-2018	ГОСТ 34.12-2018 Информационная технология. Криптографическая защита информации. Блочные шифры.
ГОСТ Р 34.13-2015	ГОСТ Р 34.13-2015 Информационная технология. Криптографическая защита информации. Режимы работы блочных шифров.
ГОСТ 34.13-2018	ГОСТ 34.13-2018 Информационная технология. Криптографическая защита информации. Режимы работы блочных шифров.
Ключ хранения	Ключ, которым производится зашифрование рабочего ключа
МДЗ	Модуль доверенной загрузки
ОС	Операционная система
ПАК	Программно-аппаратный комплекс
ПО	Программное обеспечение

Термин/ Сокращение, обозначение	**Расшифровка**
Пользовательский ключ, ключи пользователей	Ключи, хранимые на отчуждаемых носителях, которыми производится зашифрование ключа хранения
ПС	Программные средства
Рабочий ключ	Ключ, которым производится зашифрование/расшифрование информации
Р 1323565.1.026–2019	Р 1323565.1.026–2019 Информационная технология. Криптографическая защита информации. Режимы работы блочных шифров, реализующие аутентифицирующее шифрование.
Р 50.1.111-2016	Р 50.1.111-2016 Информационная технология. Криптографическая защита информации. Парольная защита ключевой информации.
Р 50.1.113-2016	Р 50.1.113-2016 Информационная технология. Криптографическая защита информации. Криптографические алгоритмы, сопутствующие применению алгоритмов электронной цифровой подписи и функции хэширования.
ТЗ	Техническое задание
ФСБ	Федеральная служба безопасности
ЭВМ	Электронно-вычислительная машина
ЭЦП	Электронная цифровая подпись

Лист регистрации изменений									
Изм.	*Номера листов (страниц)*				*Всего листов (страниц) в документе*	*Номер документа*	*Входящий номер сопроводительного документа и дата*	*Подпись*	*Дата*
	измененных	*замененных*	*новых*	*аннулированных*					