Спецификация ПО СКЗИ

Раздел "Документация"

|  |  |  |
| --- | --- | --- |
| **Обозначение** | **Наименование** | **Примечание** |
| СКЗИ-ТП-001 | Текст программы (файл: main.c) | Основной исполняемый код системы |
| СКЗИ-ОП-002 | Описание программы. Реализация шифрования/дешифрования сообщений через CRISP | Алгоритм: ГОСТ Магма (MGM\_KDF\_CMAC) |
| СКЗИ-РП-003 | Руководство пользователя. Инструкция по работе с криптографическими сообщениями | Пример: шифрование текстовых данных |
| СКЗИ-ЛОГ-004 | Формат журнала аудита (crypto\_audit.log) | Запись событий аутентификации, шифрования и очистки данных |

Раздел "Комплексы"

|  |  |  |
| --- | --- | --- |
| **Обозначение** | **Наименование** | **Примечание** |
| СКЗИ-КОМПЛЕКС-001 | Криптографический модуль обработки сообщений | Включает шифрование, дешифрование и проверку MAC |

Раздел "Компоненты"

|  |  |  |
| --- | --- | --- |
| **Обозначение** | **Наименование** | **Примечание** |
| СКЗИ-МОДУЛЬ-001 | Модуль аутентификации (authenticate\_user) | Проверка пароля "securepass" |
| СКЗИ-МОДУЛЬ-002 | Модуль логирования (log\_event) | Запись в crypto\_audit.log |
| СКЗИ-МОДУЛЬ-003 | Модуль безопасного обнуления памяти (secure\_zero) | Защита от утечек данных |
| СКЗИ-МОДУЛЬ-004 | Модуль генерации случайных данных (generate\_random) | Использование /dev/urandom |
| СКЗИ-МОДУЛЬ-005 | Модуль инициализации CRISP (crisp\_init) | Настройка контекста с ключами |
| СКЗИ-МОДУЛЬ-006 | Модуль шифрования сообщений (crisp\_create\_message) | Формирование зашифрованного сообщения |
| СКЗИ-МОДУЛЬ-007 | Модуль дешифрования сообщений (crisp\_process\_message) | Проверка MAC и расшифровка данных |
|  | Модуль освобождения ресурсов (crisp\_free\_message) | Очистка структур сообщений |

Примечания:

1. В разделе "Документация" коды видов документов соответствуют внутренней классификации.
2. Реализация шифрования соответствует ГОСТ Р 34.12-2015 ("Магма") и использует KDF на базе CMAC.
3. Блокировка доступа активируется после 5 неудачных попыток аутентификации (параметр MAX\_FAILED\_AUTH).
4. Для генерации ключей (baseKey, sourceId, keyId) используются фиксированные значения (демонстрационный режим).
5. Безопасное обнуление памяти применяется для ключей (baseKey), идентификаторов (sourceId, keyId) и временных буферов.
6. Проверка MAC выполняется при дешифровании для обеспечения целостности данных.

Описание ПО СКЗИ

а) Основные сведения о составе ПО СКЗИ

* **Состав ПО**:
  + main.c (SHA-256: *пример: 3a7b9c1d5e8f2a4b6c8d0e1f3a5b7d9*) — основной исходный файл.
  + crisp.h (SHA-256: *пример: 4c5d6e7f8a9b0c1d2e3f4a5b6c7d8e9*) — заголовочный файл криптосистемы.
  + **Зависимости**:
    - Стандартная библиотека C (libc).
    - Файл /dev/urandom для генерации случайных данных.

б) Логическая структура ПО СКЗИ

* **Модули и функции**:
  1. **Аутентификация**:
     + Проверка пароля (authenticate\_user), блокировка доступа на 10 секунд после 5 неудачных попыток.
  2. **Логирование**:
     + Запись событий в файл crypto\_audit.log (log\_event).
  3. **Генерация случайных данных**:
     + Использование /dev/urandom для заполнения буферов (generate\_random).
  4. **Криптографические операции**:
     + Инициализация контекста (crisp\_init), шифрование (crisp\_create\_message), расшифровка (crisp\_process\_message).
  5. **Безопасность**:
     + Обнуление конфиденциальных данных (secure\_zero).
* **Взаимодействие модулей**:
  1. Основная функция (main) управляет аутентификацией, инициализирует криптографический контекст, выполняет шифрование/расшифровку сообщения, логирует этапы работы.

в) Описание ПО СФ СКЗИ

* **Специальные функции**:
  + **Шифрование по ГОСТ**: Использование алгоритма Magma в режиме MGM-KDF-CMAC для шифрования данных.
  + **Управление ключами**:
    - Генерация базового ключа (baseKey), идентификаторов (sourceId, keyId).
    - Обнуление ключей после завершения работы.
  + **Целостность данных**: Проверка MAC для подтверждения корректности расшифровки.
  + **Логирование**: Фиксация всех критических операций (аутентификация, генерация ключей, ошибки).

г) Методы и правила эксплуатации

* **Требования к запуску**:
  + Наличие доступа к /dev/urandom для генерации случайных данных.
  + Установленные права на запись в файл crypto\_audit.log.
* **Ограничения**:
  + Пароль для доступа: "securepass".
  + Максимальная длина сообщения: 2048 байт.
  + Блокировка доступа при 5 неудачных попытках ввода пароля.
* **Рекомендации**:
  + Использовать ОС с поддержкой криптографических функций.
  + Регулярно обновлять ключи (baseKey, sourceId, keyId).

д) Инструкция по сборке

1. Скомпилировать программу:

gcc -o crypto\_tool main.c -O2 -Wall

1. Проверить контрольные суммы файлов (примеры из п. а).
2. Запустить исполняемый модуль:

./crypto\_tool

**Описание применения ПО СКЗИ**

а) Сведения о назначении ПО СКЗИ

Программное обеспечение (ПО) СКЗИ предназначено для обеспечения криптографической защиты данных посредством:

* Шифрования и расшифровки сообщений с использованием алгоритма ГОСТ Magma в режиме MGM (аутентифицированное шифрование).
* Генерации и управления ключами (базовый ключ, идентификаторы источника и ключа).
* Проверки целостности данных через механизм аутентификации (MAC).

б) Сведения об области применения ПО СКЗИ

Программа применяется в системах, требующих:

* Защиты конфиденциальной информации (текстовые сообщения, файлы).
* Соблюдения стандартов криптографической безопасности (ГОСТ Р 1323565.1.029-2019).
* Аудита операций (логирование событий шифрования, аутентификации, ошибок).

в) Сведения о классе решаемых ПО СКЗИ задач

Класс задач включает:

1. **Криптографические операции**:
   * Шифрование/расшифровка данных.
   * Генерация и проверка MAC для обеспечения целостности.
2. **Управление ключами**:
   * Инициализация контекста с использованием статического базового ключа.
   * Безопасное хранение и очистка ключевых материалов.
3. **Аутентификация и безопасность**:
   * Защита доступа через парольную аутентификацию.
   * Блокировка при превышении попыток ввода пароля.

г) Сведения об ограничениях при применении ПО СКЗИ

* **Аутентификация**:
  + Максимальное количество неудачных попыток ввода пароля: **5**.
  + Временная блокировка доступа: **10 секунд** при превышении лимита.
* **Технические ограничения**:
  + Использование статического ключа baseKey (требует модификации кода для динамической генерации).
  + Поддержка только POSIX-совместимых систем.

д) Сведения о минимальной конфигурации технических средств

* **Аппаратное обеспечение**:
  + Процессор с поддержкой 64-битных операций.
* **Программное обеспечение**:
  + ОС: Linux или иная POSIX-совместимая среда.
  + Библиотеки: Реализация функций crisp\_init, crisp\_create\_message, crisp\_process\_message (подключение через crisp.h).
  + Доступ к /dev/urandom для генерации случайных данных.

е) Сведения о СФ

**Системные функции (СФ)**:

1. **Аутентификация**:
   * Проверка пароля "securepass".
2. **Логирование**:
   * Запись событий в файл crypto\_audit.log (успешная аутентификация, ошибки, операции с ключами).
3. **Безопасность**:
   * Безопасное обнуление критических данных (secure\_zero).
   * Защита от перебора пароля.
4. **Криптография**:
   * Режим аутентифицированного шифрования MGM-KDF-CMAC.

**ж) Сведения о порядке работы СКЗИ**

1. **Запуск программы**:

./program

1. **Аутентификация**:
   * Ввод пароля "securepass".
   * Блокировка при 5 неудачных попытках.
2. **Инициализация контекста**:
   * Загрузка статического baseKey, sourceId, keyId.
   * Настройка режима шифрования (MGM\_KDF\_CMAC).
3. **Шифрование сообщения**:
   * Создание объекта CRISP\_Message с зашифрованным текстом.
   * Вывод метаданных (MAC, размер полезной нагрузки).
4. **Расшифровка и проверка**:
   * Обработка сообщения и проверка целостности через MAC.
5. **Очистка ресурсов**:
   * Безопасное обнуление ключей (baseKey, sourceId, keyId).
   * Освобождение памяти.

Пояснительная записка к ПО СКЗИ

а) Назначение компонентов

1. crisp.h
   * Реализация протокола CRISP (ГОСТ Р 1323565.1.029-2019).
   * Управление контекстом шифрования, создание/обработка сообщений.
2. **Основной модуль (**main**)**
   * Аутентификация пользователя.
   * Инициализация криптографического контекста.
   * Шифрование/дешифрование сообщений.
   * Логирование операций и безопасное обнуление ключевых данных.
3. **Вспомогательные функции**:
   * authenticate\_user – проверка пароля (заглушка).
   * log\_event – запись событий в аудит-лог.
   * secure\_zero – безопасное обнуление памяти.
   * generate\_random – заполнение буфера энтропией из /dev/urandom.

б) Перечень реализованных функций

1. **Аутентификация**:
   * authenticate\_user()
2. **Логирование**:
   * log\_event(const char \*msg)
3. **Безопасность**:
   * secure\_zero(void \*ptr, size\_t len)
   * generate\_random(uint8\_t \*buf, size\_t len)
4. **Работа с CRISP**:
   * crisp\_init() – инициализация контекста.
   * crisp\_create\_message() – шифрование сообщения.
   * crisp\_process\_message() – дешифрование сообщения.
   * crisp\_free\_message() – освобождение ресурсов сообщения.

в) Параметры функций

1. crisp\_init():
   * ctx – указатель на контекст;
   * baseKey[32] – мастер-ключ (256 бит);
   * sourceId[16], keyId[16] – идентификаторы источника и ключа;
   * mode – режим работы (например, MGM\_KDF\_CMAC).
2. crisp\_create\_message():
   * ctx – контекст;
   * payload – данные для шифрования;
   * payloadLen – длина данных;
   * encrypted\_msg – структура для зашифрованного сообщения.
3. crisp\_process\_message():
   * ctx – контекст;
   * encrypted\_msg – зашифрованное сообщение;
   * decrypted – буфер для расшифрованных данных;
   * decrypted\_len – длина расшифрованных данных.

г) Коды возврата

1. authenticate\_user():
   * 1 – успешная аутентификация;
   * 0 – ошибка.
2. crisp\_create\_message(), crisp\_process\_message():
   * 1 (true) – успех;
   * 0 (false) – ошибка.
3. main():
   * 0 – успешное завершение;
   * 1 – ошибка аутентификации/шифрования.

д) Экспортируемые функции

1. **Из**crisp.h:
   * crisp\_init(), crisp\_create\_message(), crisp\_process\_message(), crisp\_free\_message().
2. **Из основного модуля**:
   * log\_event() – для интеграции с внешними системами аудита.

е) Описание переменных

1. **Глобальные**:
   * failed\_attempts – счетчик неудачных попыток входа;
   * blocked\_until – время разблокировки доступа.
2. **Локальные в**main():
   * ctx – контекст CRISP;
   * baseKey[32], sourceId[16], keyId[16] – ключи и идентификаторы;
   * encrypted\_msg – структура зашифрованного сообщения;
   * decrypted – буфер для расшифрованных данных.

ж) Алгоритмы функционирования

1. **Аутентификация**:
   * Блокировка доступа на 10 секунд после 5 неудачных попыток.
2. **Шифрование**:
   * Инициализация контекста CRISP с мастер-ключом и идентификаторами.
   * Создание сообщения: шифрование данных, генерация MAC.
   * Формирование структуры CRISP\_Message (последовательный номер, MAC, payload).
3. **Дешифрование**:
   * Проверка MAC и расшифровка payload.
4. **Безопасность**:
   * Обнуление ключей и буферов через secure\_zero().

и) Критерии и методика тестирования

1. **Критерии**:
   * Корректность шифрования/дешифрования (побитовое совпадение исходных и расшифрованных данных).
   * Проверка целостности MAC.
   * Отсутствие утечек ключевых данных в памяти.
2. **Методика**:
   * Тестирование с различными входными данными (включая граничные случаи).
   * Проверка обработки ошибок (неверный пароль, некорректный MAC).
   * Анализ лога на полноту записей.
3. **Результаты**:
   * Успешное шифрование/дешифрование тестового сообщения.
   * Корректная работа блокировки доступа при 5 неудачных попыток.
   * Отсутствие остатков ключей в памяти после secure\_zero().