|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| УТВЕРЖДЕНО  Генеральный директор | СОГЛАСОВАНО  Генеральный директор |  | СОГЛАСОВАНО | УТВЕРЖДЕНО |
| Центра сертификации ФГУП НИИ «Восход»  (подпись) В.В. Петров  8.12.2024 | **ООО «ИнфоТеКС»**  (подпись) А.Б. Сидоров  8.12.2024 | Генеральный директор ООО «ИнфоТеКС»  (подпись) А.А. Васьков  18.09.2024 | Главный инженер  «ПГУПС»  (подпись) А.А. Васьков  18.09.2024 | Генеральный директор  «ГУАП»  (подпись) А.П. Духович  18.09.2024 |

**Лаборатория** ООО **«Глобал АйТи»**

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ «ПРОГРАММА»

643.22222222.11111-01 51 01- ЛУ

ЛИСТ УТВЕРЖДЕНИЯ

Листов 1

|  |  |  |
| --- | --- | --- |
|  | СОГЛАСОВАНО | СОГЛАСОВАНО |
|  | Руководитель лаборатории ООО **«Глобал АйТи»**  (подпись) А.А. Попов  8.12.2024 |  |

2024

УТВЕРЖДЕНО

643.33333333.22222-01 51 01

**Лаборатория** ООО **«Глобал АйТи»**

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ «ПРОГРАММА»

ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

643.33333333.22222-01 51 01

Листов 19

2024

**АННОТАЦИЯ**

Настоящий документ представляет собой Программу и методику испытаний Программного обеспечения, реализующего формирование и проверку электронной подписи на основе открытого коллективного ключа.

**Содержание**

[**1. ОБЪЕКТ ИСПЫТАНИЙ 2**](#_Toc180463912)

[**2. ЦЕЛЬ ИСПЫТАНИЙ 3**](#_Toc180463913)

[**3. ТРЕБОВАНИЯ К ПРОГРАММЕ 4**](#_Toc180463914)

[**4. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ 7**](#_Toc180463915)

[**5. СОСТАВ И ПОРЯДОК ИСПЫТАНИЙ 8**](#_Toc180463916)

[**6. МЕТОДЫ ИСПЫТАНИЙ 11**](#_Toc180463917)

[**7. ИНТЕГРАЛЬНАЯ ОЦЕНКА ИСПЫТАНИЙ 21**](#_Toc180463918)

[**Лист регистрации изменений 22**](#_Toc180463919)

# **ОБЪЕКТ ИСПЫТАНИЙ**

В настоящем документе Программы и методики испытаний объектом испытаний является программное обеспечение «Программа».

Полное наименование испытуемой программы: Программное обеспечение «Программа».

Сокращённое название: ПО «Программа»

Обозначение испытуемой программы: 643.33333333.22222-01 51 01

# **ЦЕЛЬ ИСПЫТАНИЙ**

Требуется провести испытания ПО «Программа» для определения его соответствия требованиям по 3 уровню контроля, представленным в руководящем документе ФСТЭК России «Защита от несанкционированного доступа к информации. Часть 1. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей» от 4 июня 1999 г. N 114

# **ТРЕБОВАНИЯ К ПРОГРАММЕ**

ПО должна исполнять следующий алгоритм:

1. **Инициализация переменных и создание ключей:**

* Генерация для каждого участника программы от 2 до 5 уникальных ключей (публичный и приватный).
* **Закрытые ключи случайны в диапазоне от 1 до 255, а открытые ключи вычисляются как удвоение закрытых ключей.**

1. **Хэширование документа:**

* Исходный документ преобразуется в хэш с использованием алгоритма xor\_hash, который применяет XOR-преобразование ко всем символам документа.

1. **Коллективная подпись документа:**

* Каждый участник последовательно подписывает хэш документа своим закрытым ключом. Итоговая подпись является результатом нескольких XOR -преобразований, выполненных в порядке участников.

1. **Проверка подписи:**

* Для проверки подлинности подписи используются публичные ключи участников в обратном порядке. В результате расшифрования должна быть получена исходная хэш-функция документа.

Программа состоит из 6-ти модулей:

1. **Модуль создания ключей (generate\_key\_pair):**

Создает случайную пару ключей (private\_key и public\_key) для участника.

1. **Модуль инициализации ключей (initialize\_key\_pairs):**

Инициализирует список пар ключей участников. Генерирует случайное количество участников в диапазоне от 2 до 5. Для каждого участника вызывает generate\_key\_pair для создания пары ключей.

1. **Модуль XOR-хэширования (xor\_hash):**

Этот модуль создает хэш документа с использованием алгоритма xor\_hash, применяя XOR-преобразование к каждому символу документа.

1. **Модуль создания подписей (create\_signatures):**

В этом модуле каждый участник последовательно подписывает хэш документа своим уникальным ключом. Формируется итоговая подпись путем последовательного применения XOR-преобразований ключей.

1. **Модуль проверки подписей (verify\_signatures):**

Этот модуль проверяет подлинность подписи, используя ключи участников в обратном порядке. Последовательно расшифровывается текущая подпись до исходного хэша документа.

1. **Основной модуль (main):**

Главная функция main() управляет этапами программы — инициализацией ключей, хэшированием документа, созданием подписей и их проверкой.

1. Требования к программной документации

Состав программной документации должен включать в себя:

1. Руководство по эксплуатации ПО «Программа»:   
   643.33333333.22222-01 91 01.
2. Спецификация ПО «Программа»: 643.33333333.22222-01 10 01.
3. Описание программы ПО «Программа»: 643.33333333.22222-01 13 01.
4. Текст программы ПО «Программа»: 643.33333333.22222-01 12 01.
5. Пояснительная записка ПО «Программа»: 643.33333333.22222-01 81 01.
6. Состав и порядок испытаний

Место проведения испытаний: ООО «ИнфоТеКС»

(Российская Федерация, г. Санкт-Петербург, Артиллерийская ул., 1Р.).

Период проведения испытаний: с 10.12.2024 по 10.01.2025.

**5.1 Технические и программные средства, используемые во время испытаний**

Для тестирования ПО должен быть выделен персональный компьютер с аппаратными характеристиками не ниже следующих:

* объем оперативной памяти 16 ГБ RAM
* свободное дисковое пространство от 128 Гб;
* Не менее 4-ядерного процессора Intel Xeon или аналогичного.

Для тестирования ПО должно быть установлено следующее ПО:

* ОС Windows 10;
* Среда Microsoft Visual Studio 2022.
* Программа PVS-Studio
* Программа для вычисления хеш-суммы и контрольной суммы файлов HashTab;
* Текстовый редактор Word 2019

**5.2 Порядок проведения испытаний**

Проверочные испытания включают в себя:

1. Проверку состава и содержания документации, представляемой Заявителем, на соответствие требованиям ГОСТ: ГОСТ 19.202-78, ГОСТ 19.402-78, ГОСТ 19.404-79, ГОСТ 19.401-78;
2. Контроль исходного состояния ПО с указанием значений контрольных сумм загрузочных модулей и исходных текстов программ, входящих в состав ПО;
3. Статистический анализ исходных текстов программ, включающий:
   * Контроль полноты и отсутствия избыточности исходных текстов ПО на уровне файлов;
   * Контроль соответствия исходных текстов ПО его объективному (загрузочному) коду;
   * Контроль полноты и отсутствия избыточности исходных текстов ПО на уровне функциональных объектов (процедур);
   * Контроль связей функциональных объектов (модулей, процедур, функций) по управлению;
   * Контроль связей функциональных объектов (модулей, процедур, функций) по информации;
   * Контроль информационных объектов различных типов;
   * Формирование перечня маршрутов выполнения функциональных объектов (процедур, функций).
4. Динамический анализ исходных текстов программ, включающий:
   * Контроль выполнения функциональных объектов (процедур, функций);
   * Сопоставление фактических маршрутов выполнения функциональных объектов (процедур, функций) и маршрутов, построенных в процессе проведения статического анализа (п. 3).
5. Отчётность по результатам проведения испытаний в форме протокола, содержащего результаты:
   * Контроля исходного состояния ПО;
   * Контроля полноты и отсутствия избыточности исходных текстов, контролируемого ПО на уровне файлов;
   * Контроля соответствия исходных текстов ПО его объектному (загрузочному) коду.
   * Контроля полноты и отсутствия избыточности исходных текстов, контролируемого ПО на уровне функциональных объектов (процедур);
   * Контроля связей функциональных объектов (модулей, процедур, функций) по управлению;
   * контроля связей функциональных объектов (модулей, процедур, функций) по информации;
   * контроля информационных объектов различных типов (например, локальных переменных, глобальных переменных, внешних переменных и т.п.);
   * формирования перечня маршрутов выполнения функциональных объектов (процедур, функций);
   * контроля выполнения функциональных объектов (процедур, функций);
   * сопоставления фактических маршрутов выполнения функциональных объектов (процедур, функций) и маршрутов, построенных в процессе проведения статического анализа.
6. Методы испытаний

**6.1 Контроль состава и содержания документации**

Проверка комплектности и состава документации производится непосредственно представителем Заказчика.

В ходе проверки производится соотнесение поставленной комплектации с заявленным перечнем программной документации, необходимой для проведения сертификационных испытаний, состав и содержание которой соответствует пунктам:

* Спецификация (ГОСТ 19.202-78), содержащая сведения о составе ПО и документации на него;
* Описание программы (ГОСТ 19.402-78), содержащее основные сведения о составе (с указанием контрольных сумм файлов, входящих в состав ПО), логической̆ структуре и среде функционирования ПО, а также описание методов, приёмов и правил эксплуатации средств технологического оснащения при создании ПО;
* Пояснительная записка (ГОСТ 19.404-79), содержащая основные сведения о назначении компонентов, входящих в состав ПО, параметрах обрабатываемых наборов данных (подсхемах баз данных), формируемых кодах возврата, описание используемых переменных, алгоритмов функционирования и т.п;
* Исходные тексты программ (ГОСТ 19.401-78), входящих в состав ПО.

Критерием положительного результата испытания считается наличие перечисленной выше документации, и её соответствие требованиям нормативных документов.

**6.2 Контроль исходного состояния ПО**

Контроль состояния исходного ПО выполняется с помощью программы HashTab для вычисления хэш-значений.

Проводится сравнение рассчитанных контрольных сумм для каждого файла, входящего в состав ПО, с контрольными суммами, приведёнными в описание программы и с эталонным образцом разработчика.

Критерием положительного результата испытания считается полное совпадение вычисленных хеш-сумм.

**6.3 Статический анализ**

**6.3.1 Контроль полноты и отсутствия избыточности исходных текстов на уровне файлов**

Полнота представленных исходных текстов на уровне файлов ПО подтверждается успешной компиляцией и сборкой исследуемого ПО. Критерием является отсутствие замечаний по успешной сборке всех представленных файлов в составе ПО, а также соответствие перечня исходных файлов с теми, которые были включены в процесс компиляции (лог компиляции).

Для контроля избыточности исходных текстов на уровне файлов определяется список всех используемых файлов. На основе анализа полученных данных выявляются избыточные файлы. Каждый избыточный файл тщательно анализируется по назначению и обоснованности его включения в состав программного обеспечения. Таким образом, обеспечивается соответствие перечня исходных файлов только тем, которые действительно необходимы для работы ПО, без лишних включений.

Критерием положительного результата испытания является соответствие заявленного в составе ПО перечня файлов с логом успешной компиляции и отсутствие файлов, которые не используются при выполнении функциональных требований ПО.

**6.3.2 Контроль соответствия исходных текстов ПО его объектному (загрузочному) коду**

Контроль соответствия исходных текстов программного обеспечения (ПО) его объектному (загрузочному) коду осуществляется методом создания загрузочных модулей из представленных исходных текстов ПО, сопоставленных с образцом разработчика. В процессе анализа учитываются возможные различия, возникающие из-за различных меток времени сборки.

Критерием положительного результата испытания является однозначное соответствие контрольных сумм файлов компиляции (загрузочного кода) с представленным образцом разработчика.

Если хеш-сумма не совпала, необходимо проанализировать hex-код исполняемого файла с помощью ПО «HxD Hex Editor». Если анализ показывает, что изменения не критичны, то необходимо указать «программы можно считать условно тождественными».

**6.3.3 Контроль связей функциональных объектов по управлению**

Производится формирование таблицы «Связи функциональных объектов по управлению», характеризующей отношения вызова и достижимости между всеми парами функциональных объектов.

Формирование таблицы:

Начиная со второго столбца необходимо указать все функциональные объекты программы. В первом столбце, начиная со второй строки также указать все функциональные объекты программы.

Если функциональный объект осуществляет вызов другого функционального объекта, то сведения о данной паре заносятся в таблицу «Связи функциональных объектов по управлению» и обозначаются символом «+». Столбец соответствует функциональному объекту, который вызывает другой функциональный объект, а строка – вызываемому объекту.

Критерием положительного результата испытания является совпадение полученных результатов с описанием, приведённым в программной документации на исследуемое программное обеспечение.

**6.3.4 Контроль связей функциональных объектов по информации**

Производится формирование таблицы «Связи функциональных объектов по информации», характеризующей процесс совместного использования одних информационных объектов в различных функциях исследуемого программного обеспечения.

Таблица «Связи функциональных объектов по информации» отображает факты совместного использования информационных объектов различными функциональными объектами.

Критерием положительного результата испытания является отсутствие в таблице «Связи функциональных объектов по информации» связей, не указанных в программной документации на поставляемое ПО.

**6.3.5 Контроль информационных объектов**

На данном этапе выполнения методики статического анализа выполняется проверка программных текстов на наличие глобальных переменных, контрольных секций и/или повторного использования переменных.

Критерием положительного результата испытания является отсутствие информационных объектов, не заявленных в предоставленной программной документации.

**6.3.6 Формирование перечня маршрутов выполнения функциональных объектов**

Сформировать граф по управлению «Маршруты выполнения функциональных объектов на уровне функций», необходимый для определения вызывающих и вызываемых функциональных объектов, на основании пунктов 6.3.3, 6.3.4, 6.3.5.

Критерием положительного результата испытания является наличие в графе всех вершин, соответствующих связям между функциональными объектами, указанными в программной документации.

**6.4 Динамический анализ исходных текстов программ**

**6.4.1 Контроль выполнения функциональных объектов**

Контроль выполнения функциональных объектов (процедур, функций) подразумевает внесение по определённому алгоритму в исходный код исследуемого программного обеспечения так называемых элементов отладочной печати («мониторов»), сборку «лабораторной» версии программного обеспечения, выполнение программного обеспечения в соответствии с официальной эксплуатационной документацией, фиксацию поведения исследуемого программного обеспечения в заявленных разработчиком режимах функционирования.

Монитор — это элемент отладочной печати, используемый для вывода информации о процессе выполнения функциональных объектов (процедур, функций) в программном обеспечении. Основными компонентами являются:

* Логгер: записывает информацию о каждом вызове функции (имя функции, параметры, временные метки).
* Обработчик данных: обрабатывает входные/выходные данные, например, проверяет корректность данных, формирует промежуточные результаты.
* Вывод информации: создает отчет или выводит информацию об изменениях в ходе выполнения.

Контроль выполнения функциональных объектов (процедур, функций) запускается отдельно для каждого исполняемого бинарного файла из состава анализируемого программного обеспечения, указанной в соответствующей таблице «Точки входа».

«Мониторы» обеспечивают выдачу во внешний файл информации о проходе контрольных точек. Совокупность информации о прохождении контрольных точек образует сведения о реальных маршрутах выполнения исследуемого программного обеспечения.

Критерием соответствия является выполнение всех заявленных в программной документации функций, реализующие функциональные требования к испытуемому ПО. Контроль выполнения определяется на основании генерируемого внешнего файла о прохождении контрольных точек.

**6.4.2 Сопоставление фактических маршрутов выполнения функциональных объектов и маршрутов, построенных в процессе проведения статического анализа**

При сопоставлении фактических маршрутов проводят анализ соответствия сведений, полученных в результате статического анализа, с данными таблицы «Маршруты выполнения функциональных объектов на уровне функций».

Критерием соответствия является совпадение результатов проверок пунктов 6.3.6. и 6.4.1.

**7 Интегральная оценка**

После проведения испытаний, требуется составить протокол об их проведении с указанием полученных результатов:

|  |  |  |
| --- | --- | --- |
| № | Наименование испытания | Соответствие 3му классу |
| 1. | Контроль состава и содержания документации |  |
| 2. | Контроль исходного состояния ПО |  |
| 3. Статический анализ исходных текстов программ | | |
| 3.1. | Контроль полноты и отсутствия избыточности исходных текстов |  |
| 3.2. | Контроль соответствия исходных текстов ПО его объектному (загрузочному) коду |  |
| 3.3. | Контроль связей функциональных объектов по управлению |  |
| 3.4. | Контроль связей функциональных объектов по информации |  |
| 3.5. | Контроль информационных объектов |  |
| 3.6. | Формирование перечня маршрутов выполнения функциональных объектов |  |
| 4. Динамический анализ исходных текстов программ | | |
| 4.1. | Контроль выполнения функциональных объектов |  |
| 4.2. | Сопоставление фактических маршрутов выполнения функциональных объектов и маршрутов, построенных в процессе проведения статического анализа |  |

Если по итогам проведенных испытаний все десять проверок имеют положительный результат, то исследуемая программа по контролю отсутствия недекларированных возможностей тестового ПО соответствует 3 уровню контроля.

|  |  |  |
| --- | --- | --- |
| Заместитель начальника  ООО «Глобал АйТи» |  | А.Б. Кепкин |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Лист регистрации изменений** | | | | | | | | | |
| Номер листов (страниц) | | | | | Всего листов (страниц) | № докум | Входящий номер сопроводит. документа и дата | Подпись | Дата |
| Изм | изменен | заменен | новых | аннулир |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |