

Министерство образования Республики Беларусь
Белорусский государственный университет
НАУЧНО-ИССЛЕДОВАТЕЛЬСКИЙ ИНСТИТУТ
ПРИКЛАДНЫХ ПРОБЛЕМ МАТЕМАТИКИ И ИНФОРМАТИКИ

УТВЕРЖДАЮ
Директор НИИ прикладных проблем
математики и информатики

Ю.С.Харин
« ____ » _____ 2022 г.

МЕТОДИКА ИСПЫТАНИЙ СРЕДСТВ КРИПТОГРАФИЧЕСКОЙ ЗАЩИТЫ
ИНФОРМАЦИИ НА СООТВЕТСТВИЕ ТРЕБОВАНИЯМ СТБ 1176.1-99

МИ.11761.10.01

Листов 21

Минск 2022

Предисловие

Настоящая методика испытаний предназначена для использования в испытательных лабораториях при проведении сертификационных испытаний средств криптографической защиты информации на соответствие требованиям СТБ 1176.1-99 «Информационная технология. Защита информации. Функция хэширования»

Содержание

1	Нормативные ссылки	4
2	Термины, обозначения и сокращения	4
3	Объект и цель испытаний	4
4	Требования к объекту испытаний	5
5	Средства и порядок испытаний	5
5.1	Общие сведения	5
5.2	Анализ программной документации	6
5.3	Тестирование	6
5.4	Анализ исходных текстов	6
6	Методы испытаний	7
6.1	Анализ программной документации	7
6.2	Тестирование	8
6.3	Анализ исходных текстов	10
	Приложение А Форма протокола анализа документации	16
	Приложение Б Форма протокола тестирования	18
	Приложение В Форма протокола анализа исходных текстов	20

1 Нормативные ссылки

В настоящем документе использованы ссылки на следующие стандарты:

ГОСТ 19.202-78 «Единая система программной документации. Спецификация. Требования к содержанию и оформлению».

ГОСТ 19.401-2000 «Единая система программной документации. Текст программы. Требования к содержанию, оформлению и контролю качества».

ГОСТ 19.402-2000 «Единая система программной документации. Описание программы. Требования к содержанию, оформлению и контролю качества».

ГОСТ 19.504-79 «Единая система программной документации. Руководство программиста. Требования к содержанию и оформлению».

СТБ 1176.1-99 «Информационная технология. Защита информации. Функция хэширования».

СТБ 34.101.31-2020 «Информационные технологии и безопасность. Алгоритмы шифрования и контроля целостности».

СТБ 34.101.77-2020 «Информационные технологии и безопасность. Криптографические алгоритмы на основе sponge-функции».

2 Термины, обозначения и сокращения

В настоящем документе применяются термины и обозначения СТБ 1176.1, а также следующие обозначения и сокращения:

Σ^n	множество всех слов длины n в алфавите Σ ;
Σ^*	множество всех слов конечной длины в алфавите Σ (включая пустое слово длины 0);
Σ^{n*}	множество всех слов из Σ^* , длина которых кратна n ;
$\dots 43210_{16}$	представление $u \in \{0, 1\}^{4*}$ шестнадцатеричным словом, октеты которого записаны справа налево (т.е. в порядке big-endian) и у которого последовательным четырем символам u соответствует один шестнадцатеричный символ (например, $10100010 = A2_{16}$);
$a \leftarrow u$	присвоение переменной a значения u ;
ЕСПД	единая система программной документации;
СКЗИ	средство криптографической защиты информации.

3 Объект и цель испытаний

На испытания представляется средство криптографической защиты информации (СКЗИ), реализующее процедуру вычисления функции хэширования СТБ 1176.1, и программная документация на СКЗИ, разработанная в соответствии с требованиями единой системы программной документации (ЕСПД).

Целью испытаний является проверка соответствия объекта испытаний требованиям СТБ 1176.1.

4 Требования к объекту испытаний

К программе объекта испытаний предъявляются следующие требования, подлежащие проверке во время проведения испытаний:

- в программе должна быть точно и полно реализовывана процедура вычисления функции хэширования СТБ 1176.1;
- программа, реализующая процедуру вычисления функции хэширования и требования СТБ 1176.1, не должна содержать недокументированные возможности.

Документация на объект испытаний должна включать документы «Спецификация», «Текст программы» и может включать документы «Описание программы», «Руководство программиста» и другие документы. Документация может быть разработана в соответствии с требованиями единой системы программной документации (ЕСПД).

5 Средства и порядок испытаний

5.1 Общие сведения

Испытания программы состоят из трех этапов:

- 1 Анализ документации.
- 2 Тестирование программы.
- 3 Анализ исходных текстов программы.

Выполнение этапа 1 осуществляется экспертами по анализу документации, выполнение этапа 2 — экспертами по тестированию, а выполнение этапа 3 — экспертами по анализу исходных текстов. К проведению испытаний должно быть привлечено не менее двух экспертов по анализу исходных текстов и один или более эксперт по тестированию. К анализу документации должен быть привлечен, по крайней мере, один эксперт по анализу исходных текстов программ.

По результатам выполнения этапа испытаний эксперт оформляет протокол результатов проверок: протокол анализа документации, протокол тестирования, протокол анализа исходных текстов. В протоколе эксперт делает вывод о соответствии (не соответствии) программы требованиям СТБ 1176.1. Примеры оформления протоколов приводятся в приложениях А, Б, В. Допускается оформления протоколов в иной форме, но с обязательным указанием результатов по каждой проводимой проверке и вывода о соответствии (не соответствии).

Если в испытываемой программе используется реализация процедуры вычисления функции хэширования СТБ 1176.1, которая в составе другой программы имеет действующий сертификат соответствия требованиям СТБ 1176.1, то проверки по тестированию и анализу исходных текстов для данной реализации могут не проводиться. При этом для подтверждения соответствия объекта испытаний требованиям СТБ 1176.1 экспертом оформляется протокол проверки совпадения контрольных характеристик (хэш-значений) файлов реализации испытываемой программы с контрольными характеристиками соответствующих файлов, указанными в сертификате соответствия.

На основании протоколов результатов проверок оформляется протокол испытаний, обобщающий результаты испытаний программы. В протоколе испытаний вывод о соответствии программы требованиям СТБ 1176.1 делается тогда и только тогда, когда вывод

о соответствии содержится во всех протоколах результатов проверок. Оформление протокола испытаний проводится в соответствии с требованиями технических нормативно-правовых актов в области сертификации продукции, а также документации, применяемой в испытательной лаборатории.

5.2 Анализ программной документации

Эксперт проводит анализ документации путем проверки соответствия документации программе объекта испытаний. Такой анализ состоит в получении экспертных заключений, касающихся проверки следующих документов:

- спецификация (см. п. 6.1.1);
- текст программы (см. п. 6.1.2);
- описание программы (см. п. 6.1.3);
- руководство программиста (см. п. 6.1.4).

Анализ документов «Описание программы» и «Руководство программиста» производится в случае их наличия.

5.3 Тестирование

Эксперт проводит тестирование путем выполнения испытываемой программы для некоторого набора проверочных входных значений и сравнения полученных результатов с истинными. Истинные результаты, используемые при тестировании, формируются с помощью эталонной реализации.

Эталонной считается реализация, которая ранее успешно прошла сертификационные испытания на соответствие СТБ 1176.1 или которая удовлетворяет следующим условиям:

1 Проведен анализ исходных текстов программ эталонной реализации. К анализу привлекались, по меньшей мере, два независимых эксперта. Использовалась методика анализа исходных текстов, определенная в п. 6.3.

2 Проведено тестирование эталонной реализации. При тестировании использовались две другие независимые реализации. Использовались тесты, определенные в п. 6.2.

Тестированию подлежит процедура вычисления функции хэширования, реализованная в программе и определенная в СТБ 1176.1.

Для организации тестирования в исходные тексты программы допускается вносить изменения и дополнения, касающиеся:

- способа чтения входных данных;
- способа записи выходных данных.

При внесении модификаций в исходные тексты должен быть проведен анализ корректности внесенных изменений.

При успешном выполнении тест возвращает признак УСПЕХ, иначе — ОШИБКА. Если при тестировании программы для некоторых входных значений получены результаты отличные от истинных значений, то эксперт по тестированию должен указать эти входные значения программы и результат ее работы, а также, по требованию, результаты промежуточных вычислений экспертам по анализу исходных текстов.

5.4 Анализ исходных текстов

Эксперт проводит анализ исходных текстов путем проверки корректности реализации в испытываемой программе процедуры вычисления функции хэширования СТБ 1176.1. Такой анализ состоит в получении экспертных заключений, касающихся:

- корректности использования локальных переменных (см. п. 6.3.1);
- корректности использования глобальных переменных (см. п. 6.3.2);
- корректности использования констант (см. п. 6.3.3);
- корректности программной логики функций программы (см. п. 6.3.4);
- корректности вызова стандартных функций (см. п. 6.3.5);
- корректности вызова функций программы (см. п. 6.3.6);
- корректности обработки исключительных ситуаций (см. п. 6.3.7);
- корректности реализации криптографических примитивов (см. п. 6.3.8);
- корректности реализации криптографических алгоритмов (см. п. 6.3.9);
- корректности управления секретными данными (см. п. 6.3.10);
- отсутствия недокументированных возможностей (см. п. 6.3.11).

6 Методы испытаний

6.1 Анализ программной документации

6.1.1 Документ «Спецификация»

При анализе документа «Спецификация» эксперт проверяет, что в нем указаны компоненты и документация, представляемые на испытания.

Если документ «Спецификация» разработан в соответствии с требованиями ЕСПД, то эксперт проверяет, что содержание и оформление документа соответствует ГОСТ 19.202.

6.1.2 Документ «Текст программы»

При анализе документа «Текст программы» эксперт проверяет, что исходные тексты программы, реализующие определенную в СТБ 1176.1 процедуру вычисления функции хэширования, представлены полностью и в виде, который использовался при сборке программы.

Если документ «Текст программы» разработан в соответствии с требованиями ЕСПД, то эксперт проверяет, что содержание и оформление документа соответствует ГОСТ 19.401.

6.1.3 Документ «Описание программы»

При анализе документа «Описание программы» эксперт проверяет выполнение следующих требований:

- в документе должна быть указана информация, однозначно идентифицирующая вызываемые стандартные функции (версия компилятора, используемые стандартные библиотеки и т.п.);
- документ должен определять программные модули, реализующие определенную в СТБ 1176.1 процедуру вычисления функции хэширования;
- описание программы в терминах программных модулей должно соответствовать исходным текстам программы.

Если документ «Описание программы» разработан в соответствии с требованиями ЕСПД, то эксперт проверяет, что содержание и оформление документа соответствует ГОСТ 19.402.

6.1.4 Документ «Руководство программиста»

При анализе документа «Руководство программиста» эксперт проверяет выполнение следующих требований:

- документ должен содержать описание всех доступных для вызова функций, реализующих определенную в СТБ 1176.1 процедуру вычисления функции хэширования;
- описание функций, реализующих определенную в СТБ 1176.1 процедуру вычисления функции хэширования, и условия их использования должны соответствовать исходным текстам программы.

При описании в документации функций должны выполняться следующие условия:

- каждая функция должна иметь описание назначения;
- каждый параметр функции должен иметь описание назначения, типа и, при необходимости, диапазона допустимых значений;
- каждая функция должна иметь описание возвращаемого результата с указанием типа;
- каждая функция должна иметь описание условий, при выполнении которых в ходе работы функции могут возникать ошибочные ситуации, требующие специальной обработки;
- в случае если при реализации криптографического алгоритма используется более одной доступной для вызова функции, должны быть указаны порядок и условия вызова данных функций.

Если документ «Руководство программиста» разработан в соответствии с требованиями ЕСПД, то эксперт проверяет, что содержание и оформление документа соответствует ГОСТ 19.504.

6.2 Тестирование

При тестировании реализации алгоритма хэширования выполняются тесты HASH.1 – HASH.5.

Входными данными тестов являются стартовое значение функции хэширования H , сообщение M и длина хэш-значения L .

В тестах для хранения результата хэширования используются слова $Y, Y' \in \{0, 1\}^L$.

Тест HASH.1

1 Задать длину хэш-значения: $L \leftarrow 256$.

2 Задать стартовое значение функции хэширования:

$$H \leftarrow \begin{array}{l} 00000000000000000000000000000000 \\ 00000000000000000000000000000000_{16}. \end{array}$$

3 Задать сообщение длины 32 октета:

$$M \leftarrow \begin{array}{l} 2931207473657428206567617373656D \\ 2065747962206F777420797472696854_{16}. \end{array}$$

4 Испытуемой реализацией вычислить хэш-значение M и сохранить результат в Y .

5 Если

$$Y = \begin{array}{l} 5C52E551F12896EC99DDC8EBB993AE4B \\ 30AC378E3FBC074C9D47A7F6B5194618_{16}, \end{array}$$

то вернуть УСПЕХ, иначе — ОШИБКА.

Тест HASH.2

- 1 Задать длину хэш-значения: $L \leftarrow 256$.
- 2 Задать стартовое значение функции хэширования:

$$H \leftarrow \begin{array}{l} 00000000000000000000000000000000 \\ 00000000000000000000000000000000_{16}. \end{array}$$

- 3 Задать сообщение длины 54 октета:

$$M \leftarrow \begin{array}{l} 6567617373656D20746962206F777420 \\ 79747269687420646572646E75682072 \\ 756F6620726F20657479622072756F66 \\ 207974666946_{16}. \end{array}$$

- 4 Испытуемой реализацией вычислить хэш-значение M и сохранить результат в Y .
- 5 Если

$$Y = \begin{array}{l} 67D3BE834B44DDE636C5C06C2209AB86 \\ 33331F250E63675FEE6C1281929AEE5D_{16}, \end{array}$$

то вернуть **УСПЕХ**, иначе — **ОШИБКА**.

Тест HASH.3

- 1 Задать длину хэш-значения: $L \leftarrow 248$.
- 2 Задать стартовое значение функции хэширования:

$$H \leftarrow \begin{array}{l} 00000000000000000000000000000000 \\ 00000000000000000000000000000000_{16}. \end{array}$$

- 3 Задать сообщение длины 54 октета:

$$M \leftarrow \begin{array}{l} 6567617373656D20746962206F777420 \\ 79747269687420646572646E75682072 \\ 756F6620726F20657479622072756F66 \\ 207974666946_{16}. \end{array}$$

- 4 Испытуемой реализацией вычислить хэш-значение M и сохранить результат в Y .
- 5 Если

$$Y = \begin{array}{l} D3BE834B44DDE636C5C06C2209AB8633 \\ 331F250E63675FEE6C1281929AEE5D_{16}, \end{array}$$

то вернуть **УСПЕХ**, иначе — **ОШИБКА**.

Тест HASH.4

- 1 Задать длину хэш-значения: $L \leftarrow 256$.
- 2 Задать стартовое значение функции хэширования:

$$H \leftarrow \begin{array}{l} \text{AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA} \\ \text{AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA}_{16}. \end{array}$$

- 3 Задать сообщение M , состоящее из 1000000 октетов 61_{16} (шестнадцатеричное представление символа 'а' в коде ASCII).
- 4 Испытуемой реализацией вычислить хэш-значение M и сохранить результат в Y .
- 5 Если

$$Y = \begin{array}{l} 80A0EBF4145859B5C4F0A2B5935AB4EE \\ B661DAB3ECFDBB959B6A020C49AF60A7_{16}, \end{array}$$

то вернуть УСПЕХ, иначе — ОШИБКА.

Тест HASH.5

- 1 Задать длину хэш-значения: $L \leftarrow 256$.
- 2 Для $i = 1, 2, \dots, 10000$ выполнить:
 - 1) псевдослучайным методом сгенерировать стартовое значение функции хеширования H ;
 - 2) псевдослучайным методом сгенерировать сообщения M длины 2048 октета;
 - 3) испытуемой реализацией вычислить хэш-значение M и сохранить результат в Y ;
 - 4) эталонной реализацией вычислить хэш-значение M и сохранить результат в Y' ;
 - 5) если $Y \neq Y'$, то вернуть ОШИБКА.
- 3 Возвратить УСПЕХ.

6.3 Анализ исходных текстов

6.3.1 Корректность использования локальных переменных

Анализ корректности использования локальных переменных проводится для всех функций программы.

Под функцией понимается часть программы, которая выполняет специфические действия и описывается типом возвращаемого значения, именем функции, формальными параметрами. Выполнение функции осуществляется посредством вызова из программы или другой функции. Данному термину в языках программирования соответствуют такие понятия как «функция», «процедура», «метод» и т.п.

Для каждой локальной переменной v функции f эксперт определяет языковые конструкции f , в которых v встречается, и выполняет следующие проверки:

- 1 При использовании v в левой части оператора присваивания тип присваиваемого значения должен совпадать с типом v , в противном случае эксперт проверяет корректность результата, учитывая стандартные правила преобразования типов, определенные в используемом языке программирования.
- 2 Перед использованием значения переменной v должна быть выполнена ее инициализация.
- 3 Обращение на чтение/запись к переменной v должно происходить в пределах установленных для нее границ, в частности, если v является переменной составного типа, то обращение к элементам v должно происходить в пределах заданных размерностей.
- 4 Если v является переменной вещественного типа, то ее использование в операциях сравнения запрещено.
- 5 Если память для v выделяется в динамической области, то перед каждым выходом из f динамическая память должна быть освобождена. После освобождения памяти не должно быть языковых конструкций, ссылающихся на нее.

Примечание — В языках программирования, снабженных средствами «сборки мусора», освобождение динамической памяти, выделяемой для локальной переменной, может быть неявным.

6.3.2 Корректность использования глобальных переменных

Для каждой глобальной переменной v эксперт определяет языковые конструкции программы, в которых v встречается. Далее выполняются проверки 1 – 4 из п. 6.3.1 и следующие проверки:

1 Если память для v выделяется в динамической области, то перед каждым выходом из программы динамическая память должна быть освобождена. После освобождения памяти не должно быть языковых конструкций, ссылающихся на нее.

2 Если v может использоваться в многопоточном режиме работы программы, то должны быть реализованы механизмы, обеспечивающие разграничение доступа к v (механизмы синхронизации доступа к глобальной переменной), при этом данные механизмы не должны блокировать доступ к v на неограниченное время.

Примечание – В языках программирования, снабженных средствами «сборки мусора», освобождение динамической памяти, выделяемой для глобальной переменной, может быть неявным.

6.3.3 Корректность использования констант

Эксперт определяет языковые конструкции программы, в которых встречаются следующие константы:

- константы C_0, C_1, C_2, C_3 (п. 5.2.1 СТБ 1176.1);
- значения коэффициентов t_{ij} (таблица 5.1 СТБ 1176.1);
- начальные значения коэффициентов v_i (таблица 5.2 СТБ 1176.1).

Для каждой языковой конструкции эксперт проверяет, что константы заданы правильно.

6.3.4 Корректность программной логики функций программы

Для каждой функции программы эксперт выполняет следующие проверки:

1 Проверка допустимости переданных параметров и используемых глобальных переменных выполняется до их использования. Проверка может не выполняться, если в документации или в комментариях к функции оговорены ограничения на входные данные, при которых функция работает правильно, и эти ограничения соблюдаются для входных данных во всех вызовах функции.

2 Все заданные варианты условных переходов возможны.

3 Все адреса безусловных переходов доступны.

4 Каждый цикл завершается за конечное число шагов, т.е. завершение цикла гарантировано.

5 После выполнения операторов функции завершение функции гарантировано: достигается одна из точек выхода из функции.

6 Отсутствуют недостижимые участки кода.

7 Цепочки последовательных действий (например, открытие файла, чтение из файла, закрытие файла) корректны. Проверка выполняется, если в функции требуется выполнить некоторое действие, требующее определенной последовательности операций.

6.3.5 Корректность вызова стандартных функций

Эксперт проверяет, что в документации, комментариях исходных текстов программ или конфигурационных файлах указана информация, однозначно идентифицирующая вызываемые стандартные функции (версии компилятора, используемых стандартных библиотек и т.п.).

Для каждого вызова стандартной функции в программе эксперт проверяет:

1 Типы и значения параметров, фактически переданных в функцию, соответствуют типам и допустимым значениям параметров функции, указанным в документации на функцию (с учетом стандартных правил преобразования типов языка программирования).

2 Если в документации на функцию указано, что функция возвращает значение, то проводится анализ корректности использования возвращаемого значения, например, корректность использования в операторе присваивания, допустимость игнорирования возвращаемого значения и т.п.

3 Если в документации на функцию указано, что вызов функции может привести к возникновению исключительной ситуации или ошибки, проверяется наличие и корректность обработки исключительной ситуации.

4 Если в документации на функцию указано, что до и после вызова функции должны выполняться определенные действия, то проверяется наличие и корректность выполнения требуемых действий.

6.3.6 Корректность вызова функций программы

Эксперт проверяет, что в документации или комментариях исходных текстов программ для каждой функции программы указана информация, определяющая:

- допустимые входные параметры и возвращаемые значения функции;
- условия, при выполнении которых в ходе работы функции могут возникать исключительные ситуации (при наличии);
- действия, которые должны выполняться до и(или) после вызова функции (при наличии).

Для каждого вызова функции программы эксперт выполняет следующие проверки:

1 Типы и значения параметров, фактически переданных в функцию, соответствуют типам и допустимым значениям параметров функции (с учетом стандартных правил преобразования типов языка программирования).

2 Если функция возвращает значение, то проводится анализ корректности использования возвращаемого значения, например, корректность использования в операторе присваивания, допустимость игнорирования возвращаемого значения и т.п.

3 Если вызов функции может привести к возникновению исключительной ситуации или ошибки, проверяется наличие и корректность обработки исключительной ситуации.

4 Если до и после вызова функции должны выполняться определенные действия, то проверяется наличие и корректность выполнения требуемых действий.

5 Если функция использует глобальные переменные, то проверяется наличие инициализации данных переменных.

6.3.7 Корректность обработки исключительных ситуаций

Под исключительной ситуацией понимается ошибочная ситуация, возникающая при выполнении программы и требующая специальной обработки. Данному термину в языках программирования соответствует такие понятия как «ошибка», «исключение» и т.п.

Для анализа корректности обработки исключительных ситуаций эксперт формирует список функций, включающий стандартные функции и функции программы, вызов которых может приводить к возникновению исключительной ситуации.

Для каждого вызова функции из составленного списка эксперт проверяет:

- 1 После каждого вызова функции имеются проверка на случай возникновения исключительной ситуации и соответствующая обработка исключительной ситуации.
- 2 При проверке и обработке исключительной ситуации учтены все возможные виды исключительных ситуаций, возникновение которых возможно для вызываемой функции.
- 3 Исключительные ситуации обрабатываются адекватно (возвращаются верные коды ошибок и сообщения об ошибках и т.п.).

6.3.8 Корректность реализации криптографических примитивов

Криптографический примитив — это определенное в СТБ 1176.1 вспомогательное преобразование, являющееся композиционной частью некоторого криптографического алгоритма.

В СТБ 1176.1 определены следующие криптографические примитивы:

- преобразования $\rho_0, \rho_1, \rho_2, \rho_3$ (п. 5.2.1 СТБ 1176.1);
- преобразование ω (п. 5.2.2 СТБ 1176.1);
- преобразование ξ (п. 5.2.3 СТБ 1176.1);
- преобразование φ (п. 5.2.4 СТБ 1176.1).

Анализируя структуру программы и используя документацию, эксперт формирует список криптографических примитивов, реализованных в программе. Для каждого примитива $g : A \rightarrow B$, осуществляющего отображение множества A в множество B , эксперт проверяет:

- наличие реализации примитива g в виде отдельной функции, части функции или композиции нескольких функций;
- тождественность реализации примитива g спецификации;
- отсутствие в g операций, не используемых для реализации примитива (наличие операций, не предусмотренных спецификацией на примитив, отражается в приложении к протоколу результатов анализа исходных текстов).

Допускается, что действие отображения g определено на множестве A^* , которое является подмножеством A . В этом случае эксперт дополнительно проверяет, что при выполнении программы прообразы отображения g всегда являются элементами A^* .

6.3.9 Корректность реализации криптографического алгоритма

В СТБ 1176.1 определен алгоритм вычисления значения функции хэширования (п. 5.5 СТБ 1176.1).

Анализируя структуру программы и используя документацию, эксперт формирует список криптографических алгоритмов, реализованных в программе. Для каждого алгоритма $f : X \times \Theta \rightarrow Y$, который ставит в соответствие входным данным $x \in X$ и параметру $\theta \in \Theta$ результат криптографического преобразования $y \in Y$, эксперт проверяет наличие соответствующей реализации алгоритма. Затем эксперт определяет множества функций реализации, в которых:

- 1) задаются параметры $\theta \in \Theta$;
- 2) задаются входные данные $x \in X$;
- 3) реализуется отображение f ;

4) возвращается результат $y \in Y$.

Данные множества функций обозначаются соответственно F_1, F_2, F_3, F_4 . Множества могут пересекаться или совпадать.

Для функций из множества F_1 эксперт проверяет корректность задания параметров $\theta \in \Theta$. При этом допустимым является использование в программном компоненте множества параметров Θ^* , которое является подмножеством Θ . Однако, использованное сужение множества Θ не должно состоять в ограничении области значений секретных параметров.

Для функций из множества F_2 эксперт проверяет корректность задания входных данных $x \in X$. При этом допускается, что множество входных данных X^* алгоритма является подмножеством X . Однако, использованное сужение множества входных данных должно быть оговорено в документации.

Примечание – Программа может обрабатывать не все допустимые входные данные. Например, могут хэшироваться сообщения только определенной длины.

Для функций из множества F_3 эксперт проверяет тождественность отображения, реализуемого функциями, спецификации на алгоритм f (при возможных ограничениях на параметры и входные данные, использованные в реализации отображения). Для этого, по результатам анализа элементов множества F_3 , составляются использованные в реализации f композиции криптографических примитивов. Затем проверяется тождественность реализованных композиций композициям криптографических примитивов, заданным в спецификации и реализующим анализируемый криптографический алгоритм. Кроме этого, эксперт проводит проверку корректности реализации вспомогательных алгоритмов, использованных в программе и не описанных в спецификации. Если такой анализ провести не удастся (алгоритм не описан в документации или описан не полно, без указания использованных источников), то по данному пункту проверки выдается отрицательное заключение по причине недостаточности данных. Если использованы простые вспомогательные алгоритмы, призванные оптимизировать выполнение программы и понятные эксперту, то их описание в документации не требуется.

Для функций из множества F_4 эксперт проверяет корректность выдачи результатов $y \in Y$ выполнения криптографического алгоритма. Сужение в реализации алгоритма f множества результатов Y является недопустимым.

6.3.10 Корректность управления секретными данными

Секретные данные — это ключи, параметры и другие данные криптографических алгоритмов, значения которых в соответствии со стандартом или документацией на СКЗИ должны быть защищены от раскрытия, т.е. должны храниться в секрете.

Секретными данными СТБ 1176.1 являются сообщения, подлежащие хэшированию, если в соответствии с документацией реализация алгоритма хэширования может использоваться для обработки критических данных (например, паролей в алгоритмах вычисления ключа по паролю или случайных данных в алгоритмах генерации псевдослучайных чисел).

Эксперт проверяет, что секретные данные используются в строгом соответствии с криптографическим алгоритмом. Допускается использование секретных данных во вспомогательных операциях с целью повышения быстродействия программной реализации криптоалгоритма. Другие операции с секретными данными не допускаются.

Эксперт проверяет, что все копии секретных данных в открытом виде уничтожаются при завершении работы с ними, при этом:

- значение секретных данных, размещенное в области памяти глобальной переменной, уничтожается перед каждым выходом из программы;
- значение секретных данных, размещенное в области памяти локальной переменной функции, уничтожается перед каждым выходом из данной функции;
- значение секретных данных, размещенное в динамической памяти, уничтожается перед каждым освобождением динамической памяти.

Примечание – Под уничтожением понимается такое изменение данных, хранящихся в электронных устройствах (оперативная память, память на магнитных носителях и др.), которое предотвращает их последующее восстановление. Например, уничтожение может состоять в записи в области памяти, занимаемой значениями секретных данных, фиксированных или случайно выбранных значений.

6.3.11 Отсутствие недокументированных возможностей

Эксперт определяет отсутствие недокументированных возможностей по результатам проверок, выполненных в п. 6.3.1 – 6.3.10.

Обнаруженные недокументированные возможности отражаются в протоколе анализа исходных текстов или в приложении к нему.

Приложение А

Форма протокола анализа документации

Экз. {Поле 1}

Протокол № {Поле 2} от {Поле 3}
результатов анализа документации
 объекта испытаний {Поле 4}, реализующего процедуру вычисления функции
 хэширования согласно СТБ 1176.1-99

1. Документы:

№	Название документа	Номер
1	{Поле 5}	{Поле 6}
2	{Поле 7}	{Поле 8}
3	{Поле 9}	{Поле 10}
4	{Поле 11}	{Поле 12}

2. При анализе документации были выполнены следующие проверки:

№	Название проверки	Отметка о выполнении
1	Проверка документа «Спецификация»	{Поле 13}
2	Проверка документа «Текст программы»	{Поле 13}
3	Проверка документа «Описание программы»	{Поле 13}
4	Проверка документа «Руководство программиста»	{Поле 13}

3. Заключение по результатам анализа документации: документация {Поле 6}, {Поле 8}, {Поле 10}, {Поле 12} соответствует (не соответствует) программе объекта испытаний в части реализации процедуры вычисления функции хэширования СТБ 1176.1-99

Эксперт,
{Поле 14}

{Поле 15}

{Поле 16}

В поле 1 указывается номер экземпляра протокола.

В поле 2 указывается номер, однозначно идентифицирующий протокол.

В поле 3 указывается дата составления протокола.

В поле 4 указывается название объекта испытаний в соответствии с представленной к испытаниям документацией.

В полях 5 и 6 указываются соответственно полное название документа «Спецификация» и его идентификационный/децимальный номер.

В полях 7 и 8 указываются соответственно полное название документа «Текст программы» и его идентификационный/децимальный номер.

В полях 9 и 10 указываются соответственно полное название документа «Описание программы» и его идентификационный/децимальный номер.

В полях 11 и 12 указываются соответственно полное название документа «Руководство программиста» и его идентификационный/децимальный номер.

В поле 13 указывается результат выполнения проверки: «положительно» — результат проверки положительный, «отрицательно» — результат проверки отрицательный. После завершения анализа документации и заполнения таблицы делается вывод о соответствии (не соответствии) документации программе объекта испытаний в части реализации процедуры вычисления функции хэширования согласно СТБ 1176.1-99.

В полях 14 и 16 указываются соответственно должность и Ф. И. О. эксперта.

В поле 15 ставится собственноручная подпись эксперта.

Информация об обнаруженных несоответствиях приводится в протоколе или приложении к протоколу в произвольной форме.

Приложение Б

Форма протокола тестирования

Экз. {Поле 1}

Протокол № {Поле 2} от {Поле 3} результатов тестирования

объекта испытаний {Поле 4}, реализующего процедуру вычисления функции
хэширования согласно СТБ 1176.1-99

1. Файлы исходных текстов программ:

№	Имя файла	Хэш-значение
1	{Поле 5}	{Поле 6}
2	{Поле 5}	{Поле 6}
...

Хэш-значения для файлов вычислены согласно {Поле 7}.

2. В ходе тестирования объекта испытаний были выполнены следующие тесты:

№	Название теста	Отметка о выполнении
1	HASH.1	{Поле 8}
2	HASH.2	{Поле 8}
...

3. Заключение по результатам тестирования: объект испытаний {Поле 4} соответствует (не соответствует) требованиям, установленным в СТБ 1176.1-99.

Эксперт,
{Поле 9}

{Поле 10}

{Поле 11}

В поле 1 указывается номер экземпляра протокола.

В поле 2 указывается номер, однозначно идентифицирующий протокол.

В поле 3 указывается дата составления протокола.

В поле 4 указывается название объекта испытаний в соответствии с представленной к испытаниям документацией.

В поле 5 указываются имена исходных файлов программ объекта испытаний.

В поле 6 указывается значение функции хэширования для тестируемых файлов, вычисленное в соответствии со стандартом, указанным в поле 7. Разрешается использовать функции хэширования, определенные в СТБ 34.101.31 или СТБ 34.101.77.

В поле 8 указывается результат выполнения теста: «положительно» — тест завершен успешно, «отрицательно» — тест завершен с ошибкой; «не проводился» — тест не проводился, так как программа не поддерживает алгоритм или режим, определенный в тесте.

После завершения тестирования и заполнения таблицы делается вывод о соответствии (не соответствии) программной реализации объекта испытаний СТБ 1176.1. Вывод о соответствии делается только тогда, когда все проводимые тесты выполнены успешно.

В полях 9, 11 указываются соответственно должность и Ф. И. О. эксперта.

В поле 10 ставится собственноручная подпись эксперта.

Приложение В

Форма протокола анализа исходных текстов

Экз. {Поле 1}

Протокол № {Поле 2} от {Поле 3}
результатов анализа исходных текстов программ
 объекта испытаний {Поле 4}, реализующего процедуру вычисления функции
 хэширования согласно СТБ 1176.1-99

1. Файлы исходных текстов программ:

№	Имя файла	Хэш-значение
1	{Поле 5}	{Поле 6}
2	{Поле 5}	{Поле 6}

Хэш-значения для файлов вычислены согласно {Поле 7}.

2. В ходе анализа исходных текстов программ были выполнены следующие проверки:

№	Название проверки	Результат проверки
1	Корректность использования локальных переменных	{Поле 8}
2	Корректность использования глобальных переменных	{Поле 8}
3	Корректность использования констант	{Поле 8}
4	Корректность программной логики функций программы	{Поле 8}
5	Корректность вызова стандартных функций	{Поле 8}
6	Корректность вызова функций программы	{Поле 8}
7	Корректность обработки исключительных ситуаций	{Поле 8}
8	Корректность реализации криптографических примитивов	{Поле 8}
9	Корректность реализации криптографических алгоритмов	{Поле 8}
10	Корректность управления секретными данными	{Поле 8}
11	Отсутствие недокументированных возможностей	{Поле 8}

3. Заключение по результатам анализа исходных текстов программ: объект испытаний {Поле 4} соответствует требованиям, установленным в СТБ 1176.1-99.

Эксперт,
{Поле 9}

{Поле 10}

{Поле 11}

В поле 1 указывается номер экземпляра протокола.

В поле 2 указывается номер, однозначно идентифицирующий протокол.

В поле 3 указывается дата составления протокола.

В поле 4 указывается название объекта испытаний в соответствии с представленной к испытаниям документацией.

В поле 5 указываются имена исходных файлов программ объекта испытаний.

В поле 6 указывается значение функции хэширования для исходных файлов программ, вычисленное в соответствии со стандартом, указанным в поле 7. Разрешается использовать функции хэширования, определенные в СТБ 34.101.31 или СТБ 34.101.77.

В поле 8 указывается результат выполнения проверки: «положительно» — результат проверки положительный, «отрицательно» — результат проверки отрицательный, «не проводилась» — проверка не требуется по причине специфики реализации программ объекта испытаний (например, в программе не используются глобальные переменные). После завершения анализа исходных текстов программ и заполнения таблицы делается вывод о соответствии (не соответствии) объекта испытаний СТБ 1176.1. Вывод о соответствии делается только тогда, когда результаты всех проводимых проверок являются положительными.

В полях 9, 11 указываются соответственно должность и Ф. И. О. эксперта.

В поле 10 ставится собственноручная подпись эксперта.

Информация об обнаруженных ошибках и недокументированных возможностях приводится в протоколе или приложении к протоколу в произвольной форме и должна включать:

- 1) описание ошибки или недокументированной возможности;
- 2) имя файла и номера строк программы, содержащих ошибку.