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 34.101.47-2017

МИ.10147.10.01

Листов 33

Минск 2022

Предисловие

Настоящая методика испытаний предназначена для использования в испытательных лабораториях при проведении сертификационных испытаний средств криптографической защиты информации на соответствие требованиям СТБ 34.101.47-2017 «Информационные технологии и безопасность. Криптографические алгоритмы генерации псевдослучайных чисел».

Содержание

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 34.101.47-2012 «Информационные технологии и безопасность. Криптографические алгоритмы генерации псевдослучайных чисел».

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 34.101.47, а также следующие сокращения:

ЕСПД единая система программной документации;

СКЗИ средство криптографической защиты информации.

3 Объект и цель испытаний

На испытания представляется средство криптографической защиты информации (СКЗИ), реализующее криптографические алгоритмы СТБ 34.101.47, и документация на СКЗИ.

Целью испытаний является проверка соответствия объекта испытаний требованиям СТБ 34.101.47.

4 Требования к объекту испытаний

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

– в программе должны быть точно и полно реализовываны криптографические алгоритмы СТБ 34.101.47, поддерживаемые объектом испытаний;

– программа, реализующая криптографические алгоритмы и требования СТБ 34.101.47, 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 34.101.47. Если программа не поддерживает некоторые алгоритмы, определенные в СТБ 34.101.47, то в протоколе делается соответствующее примечание. 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 испытуемой программе используются реализации алгоритмов СТБ 34.101.47, которые в составе других программ имеют действующие сертификаты соответствия требованиям СТБ 34.101.47, то проверки по тестированию и анализу исходных текстов для данных реализаций могут не проводиться. При этом для подтверждения соответствия объекта испытаний требованиям СТБ 34.101.47 экспертом оформляется протокол проверки совпадения контрольных характеристик (хэш-значений) файлов реализации испытуемой программы с контрольными характеристиками соответствующих файлов, указанными в сертификатах соответствия.

На основании протоколов результатов проверок оформляется протокол испытаний, обобщающий результаты испытаний программы. В протоколе испытаний вывод о соответствии программы требованиям СТБ 34.101.47 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 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 испытуемой 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 34.101.47 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, а также тестовые примеры СТБ 34.101.47.

Тестированию подлежат криптографические алгоритмы, реализованные в программе и определенные в СТБ 34.101.47, включая:

– алгоритм выработки имитовставки в режиме HMAC (см. п. 6.2.1);
– алгоритм генерации псевдослучайных чисел в режиме счетчика (см. п. 6.2.2);
– алгоритм генерации псевдослучайных чисел в режиме HMAC (см. п. 6.2.3);
– алгоритм генерации одноразовых паролей в режиме HOTP (см. п. 6.2.4);
– алгоритм генерации одноразовых паролей в режиме TOTP (см. п. 6.2.5);
– алгоритм генерации одноразовых паролей в режиме OCRA (см. п. 6.2.6).

Если программа не реализует некоторые из алгоритмов, определенных в СТБ 34.101.47, то тесты для них не выполняются.

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 испытуемой программе криптографических алгоритмов СТБ 34.101.47. 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 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 определенные в СТБ 34.101.47 криптографические алгоритмы, 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 определенные в СТБ 34.101.47 криптографические алгоритмы;

– 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 определенные в СТБ 34.101.47 криптографические алгоритмы;

– описание функций, реализующих определенные в СТБ 34.101.47 криптографические алгоритмы, 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

6.2.1 Алгоритм выработки имитовставки в режиме HMAC

При тестировании реализации алгоритма выработки имитовставки в режиме HMAC выполняются тесты HMAC.HBELT.1 – HMAC.HBELT.5. В тестах используется функция хэширования, определенная в СТБ 34.101.31.

Входными данными тестов являются ключ $K \in \{0,1\}^*$ и сообщение $X \in \{0,1\}^*$.

В тестах для хранения имитовставки X на K используются слова $Y, Y' \in \{0,1\}^{256}$.

Тест HMAC.HBELT.1

1 Задать ключ длины 29 октета:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}\ \texttt{06075316 ED247A37 39CBA383 03}_{16}.$$

2 Задать сообщение длины 32 октета:

$$X \leftarrow \texttt{BE329713 43FC9A48 A02A885F 194B09A1}\ \texttt{7ECDA4D0 1544AF8C A58450BF 66D2E88A}_{16}.$$

3 Испытуемой реализацией выработать имитовставку X на K и сохранить результат в Y.

4 Если

$$Y \leftarrow \texttt{D4828E63 12B08BB8 3C9FA653 5A463554}\ \texttt{9E411FD1 1C0D8289 359A1130 E930676B}_{16},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HMAC.HBELT.2

1 Задать ключ длины 32 октета:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}\\ \texttt{06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

2 Задать сообщение длины 32 октета:

$$X \leftarrow \texttt{BE329713 43FC9A48 A02A885F 194B09A1}\\ \texttt{7ECDA4D0 1544AF8C A58450BF 66D2E88A}_{16}.$$

3 Испытуемой реализацией выработать имитовставку X на K и сохранить результат в Y.

4 Если

$$Y \leftarrow \texttt{41FFE864 5AEC0612 E952D2CD F8DD508F}\\ \texttt{3E4A1D9B 53F6A1DB 293B19FE 76B1879F}_{16},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HMAC.HBELT.3

1 Задать ключ длины 42 октета:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}\\ \texttt{06075316 ED247A37 39CBA383 03A98BF6}\\ \texttt{92BD9B1C E5D14101 5445}_{16}.$$

2 Задать сообщение длины 32 октета:

$$X \leftarrow \texttt{BE329713 43FC9A48 A02A885F 194B09A1}\\ \texttt{7ECDA4D0 1544AF8C A58450BF 66D2E88A}_{16}.$$

3 Испытуемой реализацией выработать имитовставку X на K и сохранить результат в Y.

4 Если

$$Y \leftarrow \texttt{7D01B84D 2315C332 277B3653 D7EC6470}\\ \texttt{7EBA7CDF F7FF7007 7B1DECBD 68F2A144}_{16},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HMAC.HBELT.4

1 Для $i = 1, 2, \ldots, 10000$ выполнить:

1) псевдослучайным методом сгенерировать сообщение X длины $(j \bmod 32) + 1$ октета;

2) псевдослучайным методом сгенерировать ключ K длины $(j \bmod 64) + 1$ октета;

3) испытуемой реализацией выработать имитовставку X на K и сохранить результат в Y;

4) эталонной реализацией выработать имитовставку X на K и сохранить результат в Y';

5) если $Y \neq Y'$, то возвратить `ОШИБКА`.

2 Возвратить УСПЕХ.

6.2.2 Алгоритм генерации псевдослучайных чисел в режиме счетчика

При тестировании реализации алгоритма генерация псевдослучайных чисел в режиме счетчика выполняются тесты CTR.HBELT.1 – CTR.HBELT.3, CTR.STB11761.1 – CTR.STB11761.3. В тестах CTR.HBELT.1 – CTR.HBELT.3 используется функция хэширования, определенная в СТБ 34.101.31, а в тестах CTR.STB11761.1 – CTR.STB11761.3 — функция хэширования, определенная в СТБ 1176.1.

Входными данными тестов являются натуральное число n, ключ $K \in \{0,1\}^{256}$, синхропосылка $S \in \{0,1\}^{256}$ и дополнительные данные $X \in \{0,1\}^{256n}$. Число n определяет количество генерируемых псевдослучайных чисел.

В тестах для хранения псевдослучайных чисел, полученных на K, S и X, используются слова $Y, Y' \in \{0,1\}^{256n}$.

Тест CTR.HBELT.1

1 Задать $n \leftarrow 3$.

2 Задать ключ:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}$$
$$\texttt{06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

3 Задать синхропосылку:

$$S \leftarrow \texttt{BE329713 43FC9A48 A02A885F 194B09A1}$$
$$\texttt{7ECDA4D0 1544AF8C A58450BF 66D2E88A}_{16}.$$

4 Задать дополнительные данные:

$$X \leftarrow \begin{array}{l}\texttt{B194BAC8 0A08F53B 366D008E 584A5DE4}\\ \texttt{8504FA9D 1BB6C7AC 252E72C2 02FDCE0D}\\ \texttt{5BE3D612 17B96181 FE6786AD 716B890B}\\ \texttt{5CB0C0FF 33C356B8 35C405AE D8E07F99}\\ \texttt{E12BDC1A E28257EC 703FCCF0 95EE8DF1}\\ \texttt{C1AB7638 9FE678CA F7C6F860 D5BB9C4F}_{16}.\end{array}$$

5 Испытуемой реализацией выработать псевдослучайные числа и сохранить результат в Y.

6 Если

$$Y = \begin{array}{l}\texttt{1F66B5B8 4B733967 4533F032 9C74F218}\\ \texttt{34281FED 0732429E 0C79235F C273E269}\\ \texttt{4C0E74B2 CD5811AD 21F23DE7 E0FA742C}\\ \texttt{3ED6EC48 3C461CE1 5C33A77A A308B7D2}\\ \texttt{0F51D913 47617C20 BD4AB07A EF4F26A1}\\ \texttt{AD1362A8 F9A3D42F BE1B8E6F 1C88AAD5}_{16},\end{array}$$

то возвратить УСПЕХ, иначе — ОШИБКА.

Тест CTR.HBELT.2

1 Задать $n \leftarrow 2$.

2 Задать ключ:

$$K \leftarrow \begin{array}{l}\texttt{15987DAD 8FC01133 C1210F9F 1661D14A}\\ \texttt{F9D54F68 23C954F1 9E0212E0 04E78CF5}_{16}.\end{array}$$

3 Задать синхропосылку:

$$S \leftarrow \begin{array}{l}\texttt{FFFFFFFF FFFFFFFF 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}_{16}.\end{array}$$

4 Задать дополнительные данные:

$$X \leftarrow \begin{array}{l}\texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000000 00000000 00000000}_{16}.\end{array}$$

5 Испытуемой реализацией выработать псевдослучайные числа и сохранить результат в Y.

6 Если

$$Y = \begin{array}{l}\texttt{B010347E 3EDCE94B 241C866C 56BB5370}\\ \texttt{71296A24 20F30977 678FBD8D 858593DF}\\ \texttt{39483C67 8B38A032 68D2B317 EBAB3D18}\\ \texttt{A67BDF52 54B2E7E5 EF13C813 7E447C6C}_{16},\end{array}$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HMAC.HBELT.3

1 Задать $n \leftarrow 64$.

2 Для $i = 1, 2, \ldots, 10000$ выполнить:

1) псевдослучайным методом сгенерировать ключ K;

2) псевдослучайным методом сгенерировать синхропосылку S;

3) псевдослучайным методом сгенерировать дополнительные данные X;

4) испытуемой реализацией выработать выработать псевдослучайные числа и сохранить результат в Y;

5) эталонной реализацией выработать выработать псевдослучайные числа и сохранить результат в Y';

6) если $Y \neq Y'$, то возвратить `ОШИБКА`.

3 Возвратить `УСПЕХ`.

Тест CTR.STB11761.

1 Задать $n \leftarrow 3$.

2 Задать ключ:

$$K \leftarrow \begin{array}{l}\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}\\ \texttt{06075316 ED247A37 39CBA383 03A98BF6}_{16}.\end{array}$$

3 Задать синхропосылку:

$$S \leftarrow \begin{array}{l}\texttt{BE329713 43FC9A48 A02A885F 194B09A1}\\ \texttt{7ECDA4D0 1544AF8C A58450BF 66D2E88A}_{16}.\end{array}$$

4 Задать дополнительные данные:

$$X \leftarrow \begin{array}{l}\texttt{B194BAC8 0A08F53B 366D008E 584A5DE4}\\ \texttt{8504FA9D 1BB6C7AC 252E72C2 02FDCE0D}\\ \texttt{5BE3D612 17B96181 FE6786AD 716B890B}\\ \texttt{5CB0C0FF 33C356B8 35C405AE D8E07F99}\\ \texttt{E12BDC1A E28257EC 703FCCF0 95EE8DF1}\\ \texttt{C1AB7638 9FE678CA F7C6F860 D5BB9C4F}_{16}.\end{array}$$

5 Испытуемой реализацией выработать псевдослучайные числа и сохранить результат в Y.

6 Если

$$Y = \begin{array}{l}\texttt{F7619A01 893ED14E CA0FF583 2797086C}\\ \texttt{FB5B2F90 8CFB4B33 4BC201C9 814D744F}\\ \texttt{3F616631 B040A16E 0D1EC7A7 CC62000C}\\ \texttt{377869AD 874E473C 58493143 9FC6D41D}\\ \texttt{4DAFA372 72A93832 BA8D405F 1DC58F70}\\ \texttt{B943CAC3 3A926789 C8A3C819 E6F24F98}_{16},\end{array}$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест CTR.STB11761.2

1 Задать $n \leftarrow 1$.

2 Задать ключ:

$$K \leftarrow \begin{array}{l}\texttt{D728DC6D D56AB7F2 1459099B B0526229}\\ \texttt{022BF29B 2DC5783A 727BFB7C 8EAD8F4B}_{16}.\end{array}$$

3 Задать синхропосылку:

$$S \leftarrow \begin{array}{l}\texttt{00000000 00000000 00000000 00000000}\\ \texttt{00000000 00000001 00000000 00000001}_{16}.\end{array}$$

4 Задать дополнительные данные:

$$X \leftarrow \begin{array}{l}\texttt{3DAD3307 2312EC80 10BBCE7D 52104B46}\\ \texttt{860C2C1E 89302852 5A9B52DE 7F8D0B46}_{16}.\end{array}$$

5 Испытуемой реализацией выработать псевдослучайные числа и сохранить результат в Y.

6 Если

$$Y = \begin{array}{l}\texttt{596A3F56 871FAB68 E4E8F32D A057DF7E}\\ \texttt{EA10B288 103F64CE 634B9B59 09F339DD}_{16},\end{array}$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест CTR.STB11761.3

1 Задать $n \leftarrow 64$.
2 Для $i = 1, 2, \ldots, 10000$ выполнить:
1) псевдослучайным методом сгенерировать ключ K;
2) псевдослучайным методом сгенерировать синхропосылку S;
3) псевдослучайным методом сгенерировать дополнительные данные X;
4) испытуемой реализацией выработать выработать псевдослучайные числа и сохранить результат в Y;
5) эталонной реализацией выработать выработать псевдослучайные числа и сохранить результат в Y';
6) если $Y \neq Y'$, то возвратить `ОШИБКА`.
3 Возвратить `УСПЕХ`.

6.2.3 Алгоритм генерации псевдослучайных чисел в режиме HMAC

При тестировании реализации алгоритма генерация псевдослучайных чисел в режиме HMAC выполняются тесты HMAC.HBELT.1 – HMAC.HBELT.5. В тестах используется функция хэширования, определенная в СТБ 34.101.31.

Входными данными тестов являются натуральное число n, ключ $K \in \{0,1\}^*$ и синхропосылка $S \in \{0,1\}^*$. Число n определяет количество генерируемых псевдослучайных чисел.

В тестах для хранения псевдослучайных чисел, полученных на K и S, используются слова $Y, Y' \in \{0,1\}^{256n}$.

Тест HMAC.HBELT.1

1 Задать $n \leftarrow 3$.
2 Задать ключ длины 32 октета:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}$$
$$\texttt{06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

3 Задать синхропосылку длины 32 октета:

$$S \leftarrow \texttt{BE329713 43FC9A48 A02A885F 194B09A1}$$
$$\texttt{7ECDA4D0 1544AF8C A58450BF 66D2E88A}_{16}.$$

4 Испытуемой реализацией выработать псевдослучайные числа и сохранить результат в Y.
5 Если

$$Y = \begin{array}{l}\texttt{AF907A0E 470A3A1B 268ECCCC C0B90F23}\\ \texttt{9FE94A2D C6E01417 9FC789CB 3C3887E4}\\ \texttt{695C6B96 B84948F8 D76924E2 2260859D}\\ \texttt{B9B5FE75 7BEDA2E1 7103EE44 655A9FEF}\\ \texttt{648077CC C5002E05 61C6EF51 2C513B8C}\\ \texttt{24B4F3A1 57221CFB C1597E96 9778C1E4}_{16},\end{array}$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HMAC.HBELT.2

1 Задать $n \leftarrow 3$.

2 Задать ключ длины 29 октета:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03}_{16}.$$

3 Задать синхропосылку длины 36 октета:

$$S \leftarrow \texttt{BE329713 43FC9A48 A02A885F 194B09A1 7ECDA4D0 1544AF8C A58450BF 66D2E88A A2D74652}_{16}.$$

4 Испытуемой реализацией выработать псевдослучайные числа и сохранить результат в Y.

5 Если

$$Y = \texttt{6D935875 397CABE7 D4D32340 943A6531 2E27CC4F 692085BE 19D9420F 165D7E84 C234B746 89DE8AFB 82D6CEE5 3C508B76 B7759E8F F02C2C6D B5492C79 FC813CA2 70C4FBD6 D58B1585 61FB583C 72C5069C CBD45798 769C1D1F 46C092BB 7F5CFCB6}_{16},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HMAC.HBELT.3

1 Задать $n \leftarrow 3$.

2 Задать ключ длины 32 октета:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

3 Задать синхропосылку длины 12 октета:

$$S \leftarrow \texttt{BE329713 43FC9A48 A02A885F}_{16}.$$

4 Испытуемой реализацией выработать псевдослучайные числа и сохранить результат в Y.

5 Если

$$Y = \texttt{92D109E9 ABB13DCA 7B040325 9F2DBD21 DA37C38A 1B6CACAA 6D252A0E 6F1E6DD3 1C45D7D4 A483279F 35B7BBE2 3A4C820F 5E1CFB84 2634F429 E7E994D1 AD3F5B3F 164AD1C1 40351BAE 86C32091 7CC7A00A 8EB07D63 C49E99FC C52D6635 70AAC0C6}_{16},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HMAC.HBELT.4

1 Задать $n \leftarrow 3$.

2 Задать ключ длины 48 октета:

$$K \leftarrow \begin{array}{l}\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}\\ \texttt{06075316 ED247A37 39CBA383 03A98BF6}\\ \texttt{92BD9B1C E5D14101 5445FBC9 5E4D0EF2}_{16}.\end{array}$$

3 Задать синхропосылку длины 64 октета:

$$S \leftarrow \begin{array}{l}\texttt{BE329713 43FC9A48 A02A885F 194B09A1}\\ \texttt{7ECDA4D0 1544AF8C A58450BF 66D2E88A}\\ \texttt{A2D74652 42A8DFB3 6974C551 EB232921}\\ \texttt{D4EFD9B4 3A622875 911410EA 776CDA1D}_{16}.\end{array}$$

4 Испытуемой реализацией выработать псевдослучайные числа и сохранить результат в Y.

5 Если

$$Y = \begin{array}{l}\texttt{2AA6EDF2 9402B9BA 0AC87FEC 0F55C73A}\\ \texttt{F595B2E6 EB93765C 5684A75D B012A992}\\ \texttt{1D95B744 F8B213F4 332852AD 3041A615}\\ \texttt{40BE492E 4592BD6F 68A4EF82 E642D935}\\ \texttt{8230CD1D 96770EB3 44E0D2FD E14E2232}\\ \texttt{B3159C65 5E58D213 045BC986 BC431B18}_{16},\end{array}$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HMAC.HBELT.5

1 Задать $n \leftarrow 64$.

2 Для $i = 1, 2, \ldots, 10000$ выполнить:

1) псевдослучайным методом сгенерировать ключ K длины $(j \bmod 64) + 1$ октета;

2) псевдослучайным методом сгенерировать синхропосылку S длины $(j \bmod 32) + 1$ октета;

3) испытуемой реализацией выработать выработать псевдослучайные числа и сохранить результат в Y;

4) эталонной реализацией выработать выработать псевдослучайные числа и сохранить результат в Y';

5) если $Y \neq Y'$, то возвратить `ОШИБКА`.

3 Возвратить `УСПЕХ`.

6.2.4 Алгоритм генерации одноразовых паролей в режиме HOTP

При тестировании реализации алгоритма генерации одноразовых паролей в режиме HOTP выполняются тесты HOTP.HBELT.1 – HOTP.HBELT.4. В тестах используются функция хэширования, определенная в СТБ 34.101.31, и количество цифр в пароле $d = 8$.

Входными данными тестов являются секретный ключ $K \in \{0,1\}^{256}$ и счетчик $C \in \{0,1\}^{64}$.

В тестах для хранения одноразового пароля используются переменные $R, R' \in \{0, 1, \ldots, 10^8 - 1\}$.

Тест HOTP.HBELT.1

1 Задать ключ:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

2 Задать значение счетчика:

$$C \leftarrow \texttt{BE329713 43FC9A48}_{16}.$$

3 Испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R.

4 Если

$$R = \texttt{21157984},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HOTP.HBELT.2

1 Задать ключ:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

2 Задать значение счетчика:

$$C \leftarrow \texttt{BE329713 43FC9A49}_{16}.$$

3 Испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R.

4 Если

$$R = \texttt{17877985},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HOTP.HBELT.3

1 Задать ключ:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

2 Задать значение счетчика:

$$C \leftarrow \texttt{BE329713 43FC9A4A}_{16}.$$

3 Испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R.

4 Если

$$R = \texttt{26078636},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест HOTP.HBELT.4

1 Для $i = 1, 2, \ldots, 10000$ выполнить:
1) псевдослучайным методом сгенерировать ключ K;
2) псевдослучайным методом сгенерировать счетчик C;
3) испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R;
4) эталонной реализацией сгенерировать одноразовый пароль и сохранить результат в R';
5) если $R \neq R'$, то возвратить `ОШИБКА`.
2 Возвратить `УСПЕХ`.

6.2.5 Алгоритм генерации одноразовых паролей в режиме TOTP

При тестировании реализации алгоритма генерации одноразовых паролей в режиме TOTP выполняются тесты TOTP.HBELT.1 – TOTP.HBELT.4. В тестах используются функция хэширования, определенная в СТБ 34.101.31, и количество цифр в пароле $d = 8$.

Входными данными тестов являются секретный ключ $K \in \{0,1\}^{256}$ и округленная отметка времени T — неотрицательное целое число меньше 2^{32}.

В тестах для хранения одноразового пароля используются переменные $R, R' \in \{0, 1, \ldots, 10^8 - 1\}$.

Тест TOTP.HBELT.1

1 Задать ключ:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

2 Задать округленную отметку времени:

$$T \leftarrow \texttt{24152754}.$$

3 Испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R.
4 Если

$$R = \texttt{97660664},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест TOTP.HBELT.2

1 Задать ключ:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

2 Задать округленную отметку времени:

$$T \leftarrow \texttt{24152755}.$$

3 Испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R.
4 Если

$$R = \texttt{94431522},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест TOTP.HBELT.3

1 Задать ключ:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

2 Задать округленную отметку времени:

$$T \leftarrow \texttt{24152756}.$$

3 Испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R.
4 Если

$$R = \texttt{55973851},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест TOTP.HBELT.4

1 Для $i = 1, 2, \ldots, 10000$ выполнить:
1) псевдослучайным методом сгенерировать ключ K;
2) псевдослучайным методом сгенерировать округленную отметку времени T;
3) испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R;
4) эталонной реализацией сгенерировать одноразовый пароль и сохранить результат в R';
5) если $R \neq R'$, то возвратить `ОШИБКА`.
2 Возвратить `УСПЕХ`.

6.2.6 Алгоритм генерации одноразовых паролей в режиме OCRA

При тестировании реализации алгоритма генерации одноразовых паролей в режиме OCRA выполняются тесты OCRA.HBELT.1 – OCRA.HBELT.4. В тестах используются функция хэширования, определенная в СТБ 34.101.31 и количество цифр в пароле $d = 8$.

Входными данными тестов являются описатель $D \in \{0,1\}^{8*}$, секретный ключ $K \in \{0,1\}^{256}$, счетчик $C \in \{0,1\}^{64}$, запрос $Q \in \{0,1\}^{8*}$, округленная отметка времени T — неотрицательное целое число меньше 2^{32}, хэш-значение статического пароля клиента $P \in \{0,1\}^{256}$, идентификатор сеанса между клиентом и сервером $S \in \{0,1\}^{512}$.

В тестах для хранения одноразового пароля используются переменные $R, R' \in \{0, 1, \ldots, 10^8 - 1\}$.

Тест OCRA.HBELT.1

1 Задать описатель:

$$D \leftarrow \texttt{"OCRA-1:HOTP-HBELT-8:C-QN08-PHBELT-S064-T1M".}$$

2 Задать ключ:

$$K \leftarrow \begin{array}{l}\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}\\ \texttt{06075316 ED247A37 39CBA383 03A98BF6}_{16}.\end{array}$$

3 Задать счетчик:

$$C \leftarrow \texttt{BE329713 43FC9A4B}_{16}.$$

4 Задать запрос:

$$Q \leftarrow \texttt{"21157984".}$$

5 Задать округленную отметку времени:

$$T \leftarrow \texttt{24152759.}$$

6 Задать хэш-значение статического пароля клиента:

$$P \leftarrow \begin{array}{l}\texttt{ABEF9725 D4C5A835 97A367D1 4494CC25}\\ \texttt{42F20F65 9DDFECC9 61A3EC55 0CBA8C75}_{16}.\end{array}$$

7 Задать идентификатор сеанса между клиентом и сервером:

$$S \leftarrow \begin{array}{l}\texttt{B194BAC8 QA08F53B 366D008E 584A5DE4}\\ \texttt{8504FA9D 1BB6C7AC 252E72C2 02FDCE0D}\\ \texttt{5BE3D612 17B96181 FE6786AD 716B890B}\\ \texttt{5CB0C0FF 33C356B8 35C405AE D8E07F99}_{16}.\end{array}$$

8 Испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R.

9 Если

$$R = \texttt{77614623,}$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест OCRA.HBELT.2

1 Задать описатель:

$$D \leftarrow \texttt{"OCRA-1:HOTP-HBELT-8:C-QN08-PHBELT-S064-T1M".}$$

2 Задать ключ:

$$K \leftarrow \begin{array}{l}\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}\\ \texttt{06075316 ED247A37 39CBA383 03A98BF6}_{16}.\end{array}$$

3 Задать счетчик:

$$C \leftarrow \texttt{BE329713 43FC9A4C}_{16}.$$

4 Задать запрос:

$$Q \leftarrow \texttt{"1787798526078636".}$$

5 Задать округленную отметку времени:

$$T \leftarrow \texttt{24152759.}$$

6 Задать хэш-значение статического пароля клиента:

$$P \leftarrow \texttt{ABEF9725 D4C5A835 97A367D1 4494CC25}$$
$$\texttt{42F20F65 9DDFECC9 61A3EC55 0CBA8C75}_{16}.$$

7 Задать идентификатор сеанса между клиентом и сервером:

$$S \leftarrow \texttt{B194BAC8 QA08F53B 366D008E 584A5DE4}$$
$$\texttt{8504FA9D 1BB6C7AC 252E72C2 02FDCE0D}$$
$$\texttt{5BE3D612 17B96181 FE6786AD 716B890B}$$
$$\texttt{5CB0C0FF 33C356B8 35C405AE D8E07F99}_{16}.$$

8 Испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R.

9 Если

$$R = \texttt{99509664,}$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест OCRA.HBELT.3

1 Задать описатель:

$$D \leftarrow \texttt{"OCRA-1:HOTP-HBELT-8:C-QN08-PHBELT-S064-T1M".}$$

2 Задать ключ:

$$K \leftarrow \texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}$$
$$\texttt{06075316 ED247A37 39CBA383 03A98BF6}_{16}.$$

3 Задать счетчик:

$$C \leftarrow \texttt{BE329713 43FC9A4D}_{16}.$$

4 Задать запрос:

$$Q \leftarrow \texttt{"2607863617877985".}$$

5 Задать округленную отметку времени:

$$T \leftarrow \texttt{24152770.}$$

6 Задать хэш-значение статического пароля клиента:

$$P \leftarrow \texttt{ABEF9725 D4C5A835 97A367D1 4494CC25}$$
$$\texttt{42F20F65 9DDFECC9 61A3EC55 0CBA8C75}_{16}.$$

7 Задать идентификатор сеанса между клиентом и сервером:

$$S \leftarrow \begin{array}{l}\texttt{B194BAC8 QA08F53B 366D008E 584A5DE4}\\ \texttt{8504FA9D 1BB6C7AC 252E72C2 02FDCE0D}\\ \texttt{5BE3D612 17B96181 FE6786AD 716B890B}\\ \texttt{5CB0C0FF 33C356B8 35C405AE D8E07F99}_{16}.\end{array}$$

8 Испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R.

9 Если

$$R = \texttt{75687625},$$

то возвратить `УСПЕХ`, иначе — `ОШИБКА`.

Тест TOTP.HBELT.4

1 Задать описатель:

$$D \leftarrow \texttt{"OCRA-1:HOTP-HBELT-8:C-QN08-PHBELT-S064-T1M".}$$

2 Задать счетчик:

$$C \leftarrow \texttt{BE329713 43FC9A4B}_{16}.$$

3 Задать запрос:

$$Q \leftarrow \texttt{"21157984".}$$

4 Задать хэш-значение статического пароля клиента:

$$P \leftarrow \begin{array}{l}\texttt{ABEF9725 D4C5A835 97A367D1 4494CC25}\\ \texttt{42F20F65 9DDFECC9 61A3EC55 0CBA8C75}_{16}.\end{array}$$

5 Задать идентификатор сеанса между клиентом и сервером:

$$S \leftarrow \begin{array}{l}\texttt{B194BAC8 QA08F53B 366D008E 584A5DE4}\\ \texttt{8504FA9D 1BB6C7AC 252E72C2 02FDCE0D}\\ \texttt{5BE3D612 17B96181 FE6786AD 716B890B}\\ \texttt{5CB0C0FF 33C356B8 35C405AE D8E07F99}_{16}.\end{array}$$

6 Для $i = 1, 2, \ldots, 10000$ выполнить:

1) псевдослучайным методом сгенерировать ключ K;

2) псевдослучайным методом сгенерировать округленную отметку времени T;

3) испытуемой реализацией сгенерировать одноразовый пароль и сохранить результат в R;

4) эталонной реализацией сгенерировать одноразовый пароль и сохранить результат в R';

5) если $R \neq R'$, то возвратить `ОШИБКА`.

7 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 константы `ipad`, `opad` (п. 6.1.3 СТБ 34.101.47). 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 34.101.47 вспомогательное преобразование, являющееся композиционной частью некоторого криптографического алгоритма.

В СТБ 34.101.47 определены следующие криптографические примитивы:

– функция хэширования h (п. 5.2, 6.1.1, 6.2.1, 6.3.1, А.4 СТБ 34.101.47). Алгоритм хэширования, который определяет действие h, должен быть задан в ТНПА. Проверка должна проводиться по согласованной с Органом по сертификации методике испытаний программы, реализующей функцию хэширования согласно заданному ТНПА. Проверка может не проводиться, если реализация h уже прошла испытания по указанной методике. В таких случаях эксперт может зачесть результаты испытаний реализации h предварительно проверив совпадение испытанной ранее реализации с проверяемой;

– алгоритм выработки имитовставки в режиме HMAC `hmac`$[h]$ (п. 6.1 СТБ 34.101.47), в случае его использования в качестве вспомогательного алгоритма;

– алгоритм построения пароля по имитовставке `opt-dt` (п. А.6 СТБ 34.101.47).

Анализируя структуру программы и используя документацию, эксперт формирует список криптографических примитивов, реализованных в программе. Для каждого примитива $g : A \to B$, 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 криптографических алгоритмов

В СТБ 34.101.47 определены следующие криптографические алгоритмы:

– алгоритм выработки имитовставки в режиме HMAC (п. 6.1 СТБ 34.101.47);

– алгоритм генерации псевдослучайных чисел в режиме счетчика (п. 6.2 СТБ 34.101.47);

– алгоритм генерации псевдослучайных чисел в режиме HMAC (п. 6.3 СТБ 34.101.47);

– алгоритм генерации одноразовых паролей в режиме HOTP (п. А.7 СТБ 34.101.47);

– алгоритм генерации одноразовых паролей в режиме TOTP (п. А.8 СТБ 34.101.47);

– алгоритма генерации одноразовых паролей в режиме OCRA (п. А.9 СТБ 34.101.47).

Анализируя структуру программы и используя документацию, эксперт формирует список криптографических алгоритмов, реализованных в программе. Для каждого алгоритма $f : X \times \Theta \to Y$, 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 использоваться ключи 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 удается (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 34.101.47 являются:

– ключ K;

– псевдослучайные числа (п. 6.2.2, 6.3.2 СТБ 34.101.47), если в соответствии с документацией реализация алгоритма генерации псевдослучайных чисел может использоваться для формирования критических данных (например, секретных и личных ключей);

– значение переменной t (п. 6.1.3 СТБ 34.101.47);

– значение переменной r (п. 6.2.3 СТБ 34.101.47);

– одноразовые пароли R (п. А.6.1, А.7.1, А.9.3 СТБ 34.101.47);

– значение переменной Y (п. А.7.2, А.9.3 СТБ 34.101.47).

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 криптографические алгоритмы
согласно СТБ 34.101.47-2017

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 криптографических алгоритмов согласно СТБ 34.101.47-2017.

Эксперт, {Поле 15} {Поле 16}
{Поле 14}

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 криптографических алгоритмов согласно СТБ 34.101.47. Вывод о соответствии делается только тогда, когда результаты всех проверок являются положительными.

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 криптографические алгоритмы
согласно СТБ 34.101.47-2017

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	HMAC.HBELT.1	{Поле 8}
2	HMAC.HBELT.2	{Поле 8}
...	...	...

3. Заключение по результатам тестирования: объект испытаний {Поле 4} соответствует (не соответствует) требованиям, установленным в СТБ 34.101.47-2017.

Эксперт, {Поле 10} {Поле 11}
{Поле 9}

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 34.101.47. 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 криптографические алгоритмы
согласно СТБ 34.101.47-2017

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 34.101.47-2017.

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 34.101.47. 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.