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 ПРОГРАММЫ, РЕАЛИЗУЮЩЕЙ ПРОТОКОЛ OCSP СОГЛАСНО СТБ 34.101.26–2012

МИ.10126.10.01

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 34.101.26-2012 «Информационные технологии и безопасность. Онлайновый протокол проверки статуса сертификата (OCSP)».

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».

ГОСТ 34.973-91 (ИСО 8824-87) «Информационная технология. Взаимосвязь открытых систем. Спецификация абстрактно-синтаксической нотации версии 1 (АСН.1)».

СТБ 34.101.19-2012 «Информационные технологии и безопасность. Форматы сертификатов и списков отозванных сертификатов инфраструктуры открытых ключей».

СТБ 34.101.26-2012 «Информационные технологии и безопасность. Онлайновый протокол проверки статуса сертификата (OCSP)».

СТБ 34.101.27-2022 «Информационные технологии и безопасность. Средства криптографической защиты информации. Требования безопасности».

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

СТБ 34.101.45-2013 «Информационные технологии и безопасность. Алгоритмы электронной цифровой подписи и транспорта ключа на основе эллиптических кривых».

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 34.101.26, а также следующие сокращения:

АСН.1 абстрактно-синтаксическая нотация версии 1;

ЕСПД единая система программной документации;

СКЗИ средство криптографической защиты информации;

ТНПА технический нормативный правовой акт;

ЭЦП электронная цифровая подпись;

OCSP online certificate status protocol (онлайновый протокол проверки статуса сертификата).

3 Объект и цель испытаний

На испытания представляется средство криптографической защиты информации (СКЗИ), реализующее онлайновый протокол проверки статуса сертификата согласно СТБ 34.101.26, и документация на СКЗИ.

Целью испытаний является проверка соответствия процедур формирования и разбора запросов и ответов онлайнового протокола проверки статуса сертификата, реализованных в объекте испытаний, требованиям СТБ 34.101.26.

4 Требования к объекту испытаний

Программа объекта испытаний должна реализовывать, по крайней мере, одну из сторон протокола: OCSP-клиента или OCSP-сервера (см. п. 5 СТБ 34.101.26). Допускается реализация одной программой обеих сторон протокола.

OCSP-клиент должен поддерживать следующие процедуры протокола:

– формирование запроса;

– разбор ответа.

В свою очередь OCSP-сервер должен поддерживать следующие процедуры протокола:

– разбор запроса;

– формирование ответа.

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

– в программе должны быть точно и полно реализовываны процедуры СТБ 34.101.26, поддерживаемые объектом испытаний;

– программа, реализующая процедуры СТБ 34.101.26, 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 34.101.26. Если программа не поддерживает некоторые процедуры, определенные в СТБ 34.101.26, 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 процедур СТБ 34.101.26, которые в составе других программ имеют действующие сертификаты соответствия требованиям СТБ 34.101.26, то проверки по тестированию и анализу исходных текстов для данных реализаций могут не проводиться. При этом для подтверждения соответствия объекта испытаний требованиям СТБ 34.101.26 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 34.101.26 делается тогда и только тогда, когда вывод о соответствии содержится во всех протоколах результатов проверок. Оформление протокола испытаний проводится в соответствии с требованиями технических нормативных правовых актов (ТНПА) в области сертификации продукции, а также документации, применяемой в испытательной лаборатории.

Реализация в программе каждого криптографического алгоритма, используемого в процедурах СТБ 34.101.26, предварительно должна пройти успешные испытания по согласованной с Органом по сертификации методике испытаний.

Испытываемая программа может не поддерживать необязательный функционал, определенный в СТБ 34.101.26 (например, формирование атрибутов запроса). При этом сужение программой обязательного функционала, определенного в СТБ 34.101.26, не допускается.

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 процедур, реализованных в программе и определенных в СТБ 34.101.26, включая:

- формирование запроса (см. п. 6.2.1);
- разбор запроса (см. п. 6.2.2);
- формирование ответа (см. п. 6.2.3);
- разбор ответа (см. п. 6.2.4).

Тестирование процедуры формирования запроса/ответа выполняется путем формирования программой запросов/ответов с последующим визуальным сравнением текстового представления форматов сформированных запросов/ответов с форматами, определенными в СТБ 34.101.26.

Тестирование процедуры разбора запроса/ответа проводится путем выполнения программой разбора запросов/ответов с последующим сравнением полученных результатов с ожидаемыми.

В тестах используются запросы/ответы, представленные в виде бинарных файлов, содержащих закодированные значения типов абстрактно-синтаксической нотации версии 1 (АСН.1), спецификация которой приводится в ГОСТ 34.973. Для преобразования бинарных файлов запросов/ответов в их текстовое представление могут использоваться программы, описанные в приложении Г.1.

При успешном выполнении тест возвращает признак `УСПЕХ`, иначе — `ОШИБКА`. Если при тестировании программы для некоторых входных значений получены результаты отличные от ожидаемых, 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.

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.

5.4 Анализ исходных текстов

Эксперт проводит анализ исходных текстов путем проверки корректности реализации в испытуемой программе процедур СТБ 34.101.26. Такой анализ состоит в получении экспертных заключений, касающихся:

- корректности использования криптографических алгоритмов (см. п. 6.3.1);
- корректности управления секретными данными (см. п. 6.3.2);
- корректности процедуры формирования запроса (см. п. 6.3.3);
- корректности процедуры разбора запроса (см. п. 6.3.4);
- корректности процедуры формирования ответа (см. п. 6.3.5);
- корректности процедуры разбора ответа (см. п. 6.3.6);
- корректности обработки исключительных ситуаций (см. п. 6.3.7);
- отсутствия недокументированных возможностей (см. п. 6.3.8).

При анализе исходных текстов реализации OCSP-клиента выполняются проверки из п. 6.3.1, 6.3.3, 6.3.6 – 6.3.8. Дополнительно, при использовании OCSP-клиентом алгоритмов выработки ЭЦП выполняются проверки из п. 6.3.2.

При анализе исходных текстов реализации OCSP-сервера выполняются проверки из п. 6.3.1, 6.3.2, 6.3.4, 6.3.5, 6.3.7, 6.3.8. При выполнении данных проверок следует учитывать рекомендации по анализу исходных текстов программ, определенные в приложении В СТБ 34.101.27.

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 34.101.26 процедуры, 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 34.101.26 процедуры;

– 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 34.101.26 процедуры;

– описание функций, реализующих определенные в СТБ 34.101.26 процедуры, 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 определенной в СТБ 34.101.26 процедуры 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 Процедура формирования запроса

Входными данными, задаваемыми при тестировании процедуры формирования запроса, являются допустимые значения для параметров вызова программы, определяющих состав и содержимое компонент запроса.

В тестах для хранения запросов используются бинарные файлы, содержащие закодированных значений типов АСН.1, составляющих запрос.

При тестировании процедуры формирования запроса выполняются следующие тесты.

Тест CreateReqTest

1 Задать параметры вызова испытываемой программы, которые в соответствии с документацией необходимы для формирования запроса.
2 Средствами испытываемой программы сформировать запрос и экспортировать его на жесткий диск ПК в виде файла, содержащего закодированные значения типов АСН.1, составляющих запрос.
3 Преобразовать файл, полученный на шаге 2, в текстовое представление запроса.
4 Провести визуальный анализ текстового представления сформированного запроса на соответствие п. 6.1 СТБ 34.101.26.
5 Повторить шаги 1 – 4 не менее 9 раз, задавая различные параметры вызова испытываемой программы (если имеется такая возможность).
6 Возвратить `УСПЕХ`, если на шаге 4 при визуальном анализе запроса не выявлено несоответствий СТБ 34.101.26 (т.е. запрос является значением типа `OCSPRequest`, см. п. 6.1.1 СТБ 34.101.26), при этом:
1) формат и содержание значения компонента `tbsRequest` соответствует типу `TBSRequest` (см. п. 6.1.1 СТБ 34.101.26):

- компонент `version` содержит значение 0 типа `Version`, соответствующего типу `INTEGER` (см. приложение Б СТБ 34.101.26), или отсутствует (при задании значения по умолчанию);
- необязательный компонент `requestorName` (при его наличии) содержит значение типа `GeneralName` (см. СТБ 34.101.19) и определяет имя OCSP-клиента (данный компонент должен включаться в запрос при включении компонента `optionalSignature`);
- компонент `requestList` содержит значение типа `SEQUENCE OF` с элементами типа `Request` и определяет список проверяемых сертификатов;
- необязательный компонент `requestExtensions` (при его наличии) содержит значение типа `Extensions` (см. СТБ 34.101.19) и определяет расширения, которые могут обрабатываться OCSP-сервером (см. п. 6.3 СТБ 34.101.26);

2) формат и содержание значения необязательного компонента `optionalSignature` (при его наличии) соответствует типу `Signature` (см. п. 6.1.1 СТБ 34.101.26):

– компонент `signatureAlgorithm` содержит значение типа `AlgorithmIdentifier` (см. СТБ 34.101.19) и определяет идентификатор алгоритма ЭЦП и связанные с данным алгоритмом параметры;

– компонент `signature` содержит значение типа `BIT STRING` и определяет ЭЦП информационной части запроса;

– необязательный компонент `certs` (при его наличии) содержит значение типа `SEQUENCE OF` с элементами типа `Certificate` (см. п. 6.1 СТБ 34.101.19) и определяет цепочку сертификатов, демонстрирующих действительность открытого ключа ЭЦП;

3) формат и содержание каждого элемента компонента `requestList` соответствует типу `Request` (см. п. 6.1.1 СТБ 34.101.26):

– компонент `reqCert` содержит значение типа `CertID` и определяет проверяемый сертификат;

– необязательный компоненте `singleRequestExtensions` (при его наличии) содержит значение типа `Extensions` (см. СТБ 34.101.19) и определяет расширения, которые относятся к проверяемому сертификату и которые могут обрабатываться OCSP-сервером (см. п. 6.3 СТБ 34.101.26);

4) формат и содержание компонента `reqCert` соответствует типу `CertID` (см. п. 6.1.1 СТБ 34.101.26):

– компонент `hashAlgorithm` содержит значение типа `AlgorithmIdentifier` (см. СТБ 34.101.19) и определяет идентификатор алгоритма хэширования, применяемого для определения значений компонентов `issuerNameHash` и `issuerKeyHash`, и связанные с данным алгоритмом параметры;

– компонент `issuerNameHash` содержит значение типа `OCTET STRING` и определяет хэш-значение уникального имени эмитента (в СТБ 34.101.19 имя эмитента представляется значением типа Name, хэшируется кодовое представление данного значения, для кодирования должны использоваться отличительные правила);

– компонент `issuerKeyHash` содержит значение типа `OCTET STRING` и определяет хэш-значение открытого ключа эмитента (в СТБ 34.101.19 открытый ключ представляется значением типа `SubjectPublicKeyInfo`, хэшируется кодовое представление значения компонента `subjectPublicKey` данного типа, за исключением тега и длины);

– компонент `serialNumber` содержит значение типа `CertificateSerialNumber` (см. СТБ 34.101.19) и определяет серийный номер сертификата, информация о статусе которого запрашивается.

7 Возвратить `ОШИБКА`.

Примечание — Задаваемые в тесте CreateReqTest параметры вызова испытываемой программы в совокупности должны покрывать наиболее полный функционал, предоставляемый испытываемой программой по формированию запроса.

6.2.2 Процедура разбора запроса

Входными данными, задаваемыми при тестировании процедуры разбора запроса, являются запросы в виде бинарных файлов, содержащих закодированные значения типов АСН.1, составляющих запрос.

При тестировании процедуры разбора запроса выполняются базовые тесты и тесты известного ответа. Базовые тесты являются обязательными. Тесты известного ответа являются дополнительными и предназначены для разбора запросов на проверку статуса сертификата открытого ключа алгоритмов подписи и транспорта ключа, определенных в СТБ 34.101.45.

Для базовых тестов запросы предоставляются совместно с испытываемой программой или формируются экспертом (при наличии необходимого программного обеспечения). Перед тестированием процедуры разбора запроса эксперт проводит визуальное сравнение текстового представления запроса с форматом, определенном в СТБ 34.101.26. Для изменения запросов, которое выполняется в некоторых базовых тестах, могут использоваться программы, описанные в приложении Г.2.

Для тестов известного ответа в качестве входных данных используются запросы, определенные в приложении Д (закодированные АСН.1-файлы с данными запросами являются неотъемлемой частью настоящей методики).

Базовые тесты. В ходе тестирования процедуры разбора запроса выполняются следующие базовые тесты.

Тест ValidReqTest

1 Задать корректный запрос на проверку статуса сертификата.
2 Средствами испытываемой программы выполнить разбор запроса.
3 Повторить шаги 1 – 2 не менее 9 раз, задавая различные корректные запросы на проверку статуса сертификата.
4 Возвратить `УСПЕХ`, если на шаге 2 запрос успешно обработан и признан корректным.
5 Возвратить `ОШИБКА`.

Примечание — Задаваемые в тесте ValidReqTest запросы должны покрывать наиболее полный функционал, предоставляемый испытываемой программой по разбору запросов на проверку статуса сертификата.

Тест InvalidReqTest

1 Изменить запрос на проверку статуса сертификата таким образом, чтобы запрос стал некорректным.
2 Средствами испытываемой программы выполнить разбор измененного запроса.
3 Повторить шаги 1 – 2 не менее 9 раз, задавая различные измененные запросы на проверку статуса сертификата.
4 Возвратить `УСПЕХ`, если на шаге 2 запрос отклонен и возвращена ошибка, соответствующая внесенным изменениям.
5 Возвратить `ОШИБКА`.

Примечание 1 — Изменение запроса может состоять, например, в изменении значения компонента `signature`.

Примечание 2 — Задаваемые в тесте InvalidReqTest запросы должны покрывать наиболее полный функционал, предоставляемый испытываемой программой по разбору запросов на проверку статуса сертификата.

Тесты известного ответа. В ходе тестирования процедуры разбора запроса выполняются следующие тесты известного ответа.

Тест VReqTest1

1 Задать в качестве запроса на проверку статуса сертификата файл «VReqTest1.der».
2 Средствами испытываемой программы выполнить процедуру разбора запроса.
3 Возвратить `УСПЕХ`, если запрос успешно обработан.
4 Возвратить `ОШИБКА`.

Примечание — Цель теста VReqTest1 — проверка способности реализации обрабатывать корректные значения компонент в запросе.

Тест VReqTest2

1 Задать в качестве запроса на проверку статуса сертификата файл «VReqTest2.der».
2 Средствами испытываемой программы выполнить процедуру разбора запроса.
3 Возвратить `УСПЕХ`, если запрос успешно обработан.
4 Возвратить `ОШИБКА`.

Примечание — Цель теста VReqTest2 — проверка способности реализации обрабатывать корректные значения компонент в запросе, содержащем подпись отправителя (компонент `signature`).

Тест VReqTest3

1 Задать в качестве запроса на проверку статуса сертификата файл «VReqTest3.der».
2 Средствами испытываемой программы выполнить процедуру разбора запроса.
3 Возвратить `УСПЕХ`, если запрос успешно обработан.
4 Возвратить `ОШИБКА`.

Примечание — Цель теста VReqTest3 — проверка способности реализации обрабатывать корректные значения компонент в запросе, содержащем информацию о двух сертификатах, статус которых требуется проверить.

Тест ISignReqTest4

1 Задать в качестве запроса на проверку статуса сертификата файл «ISignReqTest4.der».
2 Средствами испытываемой программы выполнить процедуру разбора запроса.
3 Возвратить `УСПЕХ`, если запрос отклонен.
4 Возвратить `ОШИБКА`.

Примечание — Запрос содержит некорректное значение подписи. Цель теста ISignReqTest4 — проверка способности реализации выявлять некорректные подписи в запросах.

Тест IVerReqTest5

1 Задать в качестве запроса на проверку статуса сертификата файл «IVerReqTest5.der».
2 Средствами испытываемой программы выполнить процедуру разбора запроса.
3 Возвратить `УСПЕХ`, если запрос отклонен.
4 Возвратить `ОШИБКА`.

Примечание — Запрос содержит некорректное значение версии запроса. Цель теста IVerReqTest5 — проверка способности реализации выявлять некорректные значения версии запроса.

Тест IDigAlgIDReqTest6

1 Задать в качестве запроса на проверку статуса сертификата файл «IDigAlgIDReqTest6.der».
2 Средствами испытываемой программы выполнить процедуру разбора запроса.
3 Возвратить `УСПЕХ`, если запрос отклонен.
4 Возвратить `ОШИБКА`.

Примечание — Запрос содержит некорректное значение идентификатора алгоритма хэширования. Цель теста IDigAlgIDReqTest6 — проверка способности реализации выявлять некорректные значения идентификатора алгоритма хэширования.

Тест INameHashReqTest7

1 Задать в качестве запроса на проверку статуса сертификата файл «INameHashReqTest7.der».
2 Средствами испытываемой программы выполнить процедуру разбора запроса.
3 Возвратить `УСПЕХ`, если запрос отклонен.
4 Возвратить `ОШИБКА`.

Примечание — Цель теста INameHashReqTest7 — проверка способности реализации выявлять в запросе хэш-значения уникального имени эмитента, которые имеют некорректную длину.

Тест IKeyHashReqTest8

1 Задать в качестве запроса на проверку статуса сертификата файл «IKeyHashReqTest8.der».
2 Средствами испытываемой программы выполнить процедуру разбора запроса.
3 Возвратить `УСПЕХ`, если запрос отклонен.
4 Возвратить `ОШИБКА`.

Примечание — Цель теста INameHashReqTest8 — проверка способности реализации выявлять в запросе хэш-значения открытого ключа эмитента, которые имеют некорректную длину.

6.2.3 Процедура формирования ответа

Входными данными, задаваемыми при тестировании процедуры формирования ответа, являются допустимые значения для параметров вызова программы, определяющих состав и содержимое компонент ответа.

В тестах для хранения ответов используются бинарные файлы, содержащие закодированных значений типов АСН.1, составляющих ответ.

При тестировании процедуры формирования ответа выполняются следующие тесты.

Тест CreateRespTest

1 Задать параметры вызова испытываемой программы, которые в соответствии с документацией необходимы для формирования ответа.
2 Средствами испытываемой программы сформировать ответ и экспортировать его на жесткий диск ПК в виде файла, содержащего закодированные значения типов АСН.1, составляющих ответ.
3 Преобразовать файл, полученный на шаге 2, в текстовое представление ответа.
4 Провести визуальный анализ текстового представления сформированного ответа на соответствие п. 6.2 СТБ 34.101.26.
5 Повторить шаги 1 – 4 не менее 9 раз, задавая различные параметры вызова испытываемой программы (если имеется такая возможность).
6 Возвратить `УСПЕХ`, если на шаге 4 при визуальном анализе ответа не выявлено несоответствий СТБ 34.101.26 (т.е. ответ является значением типа OCSPResponse, см. п. 6.2.1 СТБ 34.101.26), при этом:
1) формат компонента `responseStatus` соответствует типу `OCSPResponseStatus` (см. п. 6.2.1 СТБ 34.101.26) и содержит одно из значений перечисления `ENUMERATED` (см. п. 5.4 СТБ 34.101.26):

- значение `successful` (0) в случае, если ответ действителен;
- значение `malformedRequest` (1) в случае, если синтаксис полученного запроса не соответствует синтаксису OCSP;
- значение `internalError` (2) в случае, если возникло недопустимое внутреннее состояние OCSP-сервере;
- значение `tryLater` (3) в случае, если OCSP-сервер находится в рабочем состоянии, но временно не может предоставить информацию о статусе запрошенного сертификата;
- значение `sigRequired` (5) в случае, если для получения информации о статусе запрошенного сертификата требуется, чтобы запрос был подписан OCSP-клиентом;
- значение `unauthorized` (6) в случае, если OCSP-клиент не имеет полномочий делать запрос на данный OCSP-сервер;

2) формат и содержание значения необязательного компонента `responseBytes` (должен присутствовать, только если компонент `responseStatus` принимает значение `successful`) соответствует типу `ResponseBytes` (см. п. 6.2.1 СТБ 34.101.26):

- компонент `responseType` содержит значение `id-pkix-ocsp-basic` (см. приложение Б СТБ 34.101.26) типа `AlgorithmIdentifier` (см. СТБ 34.101.19) и определяет основной тип ответа;

– компонент `response` содержит значение типа `OCTET STRING` и определяет ответ;

3) формат и содержание компонента `response` соответствует закодированному по отличительным правилам значению типа `BasicOCSPResponse` (см. п. 6.2.1 СТБ 34.101.26):

– компонент `tbsResponseData` содержит значение типа `ResponseData` и определяет ответы по всем запрашиваемым сертификатам;

– компонент `signatureAlgorithm` содержит значение типа `AlgorithmIdentifier` (см. СТБ 34.101.19) и определяет идентификатор алгоритма ЭЦП и связанные с данным алгоритмом параметры;

– компонент `signature` содержит значение типа `AlgorithmIdentifier` (см. СТБ 34.101.19) и определяет ЭЦП, вычисленная от хэш-значения компонента `tbsResponseData` (компонент `tbsResponseData` ДОЛЖЕН быть предварительно закодирован с использованием отличительных правил кодирования);

– необязательный компонент `certs` (при его наличии) содержит значение типа `Certificate` (см. СТБ 34.101.19) и определяет цепочку сертификатов, демонстрирующих действительность открытого ключа ЭЦП;

4) формат и содержание компонента `tbsResponseData` соответствует типу `ResponseData` (см. п. 6.2.1 СТБ 34.101.26):

– компонент `version` содержит значение 0 типа `Version`, соответствующего типу `INTEGER` (см. приложение Б СТБ 34.101.26), или отсутствует (при задании значения по умолчанию);

– компонент `responderID` содержит значение типа `ResponderID` и определяет OCSP-сервер с использованием значения типа `Name` (см. СТБ 34.101.19), которое непосредственно задает имя сервера, или значения типа OCTET STRING; которое является хэш-значением от кодового представления (за исключением тега и длины) значения типа `SubjectPublicKeyInfo` (для кодирования должны использоваться отличительные правила, а в качестве алгоритма хэширования должен использоваться алгоритм, определенный в запросе);

– компонент `producedAt` содержит значение типа `GeneralizedTime` (см. СТБ 34.101.19) и определяет время создания ответа (см. 5.5 СТБ 34.101.26);

– компонент `responses` содержит значение типа `SEQUENCE OF` с элементами типа `SingleResponse` и определяет список ответов по всем проверяемым сертификатам;

– необязательный компонент `responseExtensions` (при его наличии) содержит значение типа `Extensions` (см. СТБ 34.101.19) и определяет необязательные расширения, которые могут обрабатываться OCSP-клиентом (см. п. 6.3 СТБ 34.101.26).

5) формат и содержание каждого элемента компонента responses соответствует типу `SingleResponse` (см. п. 6.2.1 СТБ 34.101.26):

– компонент `certID` содержит значение типа `CertID` (см. п. 6.1.1 СТБ 34.101.26) и определяет проверяемый сертификат;

– компонент `certStatus` содержит значение типа `CertStatus` и определяет статус сертификата (см. п. 5.3 СТБ 34.101.26);

- компонент `thisUpdate` и необязательный компонент `nextUpdate` (при его наличии) содержан значения типа `GeneralizedTime` (см. СТБ 34.101.19) и определяют интервал действия ответа (см. п. 5.5 СТБ 34.101.26);
- необязательный компонент `singleExtensions` (при его наличии) содержит значение типа `Extensions` (см. СТБ 34.101.19) и определяет необязательные расширения, которые могут обрабатываться OCSP-клиентом и которые содержат дополнительную информацию о проверяемом сертификате (см. п. 6.3 СТБ 34.101.26);

6) формат и содержание компонента `certID` соответствует типу `CertID` (см. п. 6.1.1 СТБ 34.101.26):

- компонент `hashAlgorithm` содержит значение типа `AlgorithmIdentifier` (см. СТБ 34.101.19) и определяет идентификатор алгоритма хэширования, который используется для определения значений компонентов `issuerNameHash` и `issuerKeyHash`, и связанные с данным алгоритмом параметры;
- компонент `issuerNameHash` содержит значение типа `OCTET STRING` и определяет хэш-значение уникального имени эмитента (в СТБ 34.101.19 имя эмитента представляется значением типа `Name`, хэшируется кодовое представление данного значения, для кодирования должны использоваться отличительные правила);
- компонент `issuerKeyHash` содержит значение типа `OCTET STRING` и определяет хэш-значение открытого ключа эмитента (в СТБ 34.101.19 открытый ключ представляется значением типа `SubjectPublicKeyInfo`, хэшируется кодовое представление значения компонента `subjectPublicKey` данного типа, за исключением тега и длины);
- компонент `serialNumber` содержит значение типа `CertificateSerialNumber` (см. СТБ 34.101.19) и определяет серийный номер сертификата, информация о статусе которого предоставляется;

7) формат и содержание компонента `certStatus` соответствует типу `CertStatus`, который является выбором одного из следующих компонентов (см. п. 5.3, 6.2.1 СТБ 34.101.26):

- компонент `good` содержит значение типа `NULL` и означает, что сертификат не отозван;
- компонент `revoked` содержит значение типа `RevokedInfo` и означает, что сертификат был отозван;
- компонент `unknown` содержит значение типа `UnknownInfo`, соответствующего типу `NULL`, и означает, что OCSP-серверу ничего не известно про запрашиваемый сертификат;

8) формат и содержание компонента `revoked` (при его наличии) соответствует типу `RevokedInfo` (см. п. 6.2.1 СТБ 34.101.26):

- компонент `revocationTime` содержит значение типа `GeneralizedTime` и определяет время, когда проверяемый сертификат был отозван или его действие было временно приостановлено;
- необязательный компонент `revocationReason` содержит значение типа `CRLReason` (см. СТБ 34.101.19) и определяет указывает причину отзыва сертификата.

7 Возвратить `ОШИБКА`.

Примечание — Задаваемые в тесте CreateRespTest параметры вызова испытываемой программы в совокупности должны покрывать наиболее полный функционал, предоставляемый испытываемой программой по формированию ответа.

6.2.4 Процедура разбора ответа

Входными данными, задаваемыми при тестировании процедуры разбора ответа, являются ответы в виде бинарных файлов, содержащих закодированные значения типов АСН.1, составляющих ответ.

При тестировании процедуры разбора ответа выполняются базовые тесты и тесты известного ответа. Базовые тесты являются обязательными. Тесты известного ответа являются дополнительными и предназначены для разбора ответов со статусом сертификата открытого ключа алгоритмов подписи и транспорта ключа, определенных в СТБ 34.101.45.

Для базовых тестов ответы предоставляются совместно с испытываемой программой или формируются экспертом (при наличии необходимого программного обеспечения). Перед тестированием процедуры разбора ответа эксперт проводит визуальное сравнение текстового представления ответа с форматом, определенном в СТБ 34.101.26. Для изменения ответов, которое выполняется в некоторых базовых тестах, могут использоваться программы, описанные в приложении Г.2.

Для тестов известного ответа в качестве входных данных используются ответы, определенные в приложении Д (закодированные АСН.1-файлы с данными ответами являются неотъемлемой частью настоящей методики).

Базовые тесты. В ходе тестирования процедуры разбора ответа выполняются следующие базовые тесты.

Тест ValidRespTest

1 Задать корректный ответ со статусом сертификата.
2 Средствами испытываемой программы выполнить разбор ответа.
3 Повторить шаги 1 – 2 не менее 9 раз, задавая различные корректные ответы со статусом сертификата.
4 Возвратить `УСПЕХ`, если на шаге 2 ответ успешно обработан и признан корректным.
5 Возвратить `ОШИБКА`.

Примечание — Задаваемые в тесте ValidRespTest ответы должны покрывать наиболее полный функционал, предоставляемый испытываемой программой по разбору ответов со статусом сертификата.

Тест InvalidRespTest

1 Изменить ответ со статусом сертификата таким образом, чтобы ответ стал некорректным.
2 Средствами испытываемой программы выполнить разбор измененного ответа.
3 Повторить шаги 1 – 2 не менее 9 раз, задавая различные измененные ответы со статусом сертификата.
4 Возвратить `УСПЕХ`, если на шаге 2 ответ отклонен и возвращена ошибка, соответствующая внесенным изменениям.

5 Возвратить `ОШИБКА`.

Примечание 1 — Изменение ответа может состоять, например, в изменении значения компонента `signature`.

Примечание 2 — Задаваемые в тесте InvalidRespTest ответы должны покрывать наиболее полный функционал, предоставляемый испытываемой программой по разбору ответов со статусом сертификата.

Тесты известного ответа. В ходе тестирования процедуры разбора ответа выполняются следующие тесты известного ответа.

Тест VRespTest1

1 Задать в качестве ответа со статусом сертификата файл «VRespTest1.der».
2 Средствами испытываемой программы выполнить процедуру разбора ответа.
3 Возвратить `УСПЕХ`, если ответ успешно обработан.
4 Возвратить `ОШИБКА`.

Примечание — Цель теста VRespTest1 — проверка способности реализации обрабатывать корректные значения компонент в ответе.

Тест VRespTest2

1 Задать в качестве ответа со статусом сертификата файл «VRespTest2.der».
2 Средствами испытываемой программы выполнить процедуру разбора ответа.
3 Возвратить `УСПЕХ`, если ответ успешно обработан.
4 Возвратить `ОШИБКА`.

Примечание — Цель теста VRespTest2 — проверка способности реализации обрабатывать корректные значения компонент в ответе, содержащем информацию о статусе двух сертификатов.

Тест VRespTest3

1 Задать в качестве ответа со статусом сертификата файл «VRespTest3.der».
2 Средствами испытываемой программы выполнить процедуру разбора ответа.
3 Возвратить `УСПЕХ`, если ответ успешно обработан.
4 Возвратить `ОШИБКА`.

Примечание — Цель теста VRespTest3 — проверка способности реализации обрабатывать корректные значения компонент в ответе, содержащем информацию об отозванном сертификате.

Тест IStatusRespTest4

1 Задать в качестве ответа со статусом сертификата файл «IStatusRespTest4.der».
2 Средствами испытываемой программы выполнить процедуру разбора ответа.
3 Возвратить `УСПЕХ`, если ответ отклонен.
4 Возвратить `ОШИБКА`.

Примечание — Цель теста IStatusRespTest4 — проверка способности реализации обрабатывать некорректные значения компонента `responseStatus` со статусом ответа.

Тест ISignRespTest5

1 Задать в качестве ответа со статусом сертификата файл «ISignRespTest5.der».
2 Средствами испытываемой программы выполнить процедуру разбора ответа.
3 Возвратить `УСПЕХ`, если ответ отклонен.
4 Возвратить `ОШИБКА`.

Примечание — Ответ содержит некорректное значение подписи. Цель теста ISignRespTest5 — проверка способности реализации выявлять некорректные подписи в ответах.

Тест ITypeRespTest6

1 Задать в качестве ответа со статусом сертификата файл «ITypeRespTest6.der».
2 Средствами испытываемой программы выполнить процедуру разбора ответа.
3 Возвратить `УСПЕХ`, если ответ отклонен.
4 Возвратить `ОШИБКА`.

Примечание — Цель теста ITypeRespTest6 — проверка способности реализации обрабатывать некорректные значения компонента `responseType`.

Тест ISignAlgIDRespTest7

1 Задать в качестве запроса на проверку статуса сертификата файл «ISignAlgIDRespTest7.der».
2 Средствами испытываемой программы выполнить процедуру разбора запроса.
3 Возвратить `УСПЕХ`, если запрос отклонен.
4 Возвратить `ОШИБКА`.

Примечание — Запрос содержит некорректное значение идентификатора алгоритма подписи. Цель теста ISignAlgIDRespTest7 — проверка способности реализации выявлять некорректные значения идентификатора алгоритма подписи.

6.3 Анализ исходных текстов

6.3.1 Корректность использования криптографических алгоритмов

В процедурах, реализуемых OCSP-клиентом и OCSP-сервером, используются алгоритмы хэширования и алгоритмы электронной цифровой подписи.

Использование функций, реализующих криптографический алгоритм, должно выполняться в соответствии с СТБ 34.101.26, ТНПА на криптографический алгоритм и документацией на испытываемую программу. Для каждого вызова в программе функций, реализующих криптографический алгоритм, 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.

Примечание — 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.

6.3.2 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.

В процедуре формирования запроса/ответа для подписи данных используются следующие секретные данные:

– личный ключ подписи;

– одноразовый ключ подписи.

Эксперт проверяет, что секретные данные используются в строгом соответствии с криптографическим алгоритмом. 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.

6.3.3 Корректность процедуры формирования запроса

При анализе корректности реализации процедуры формирования запроса исходные тексты программы оцениваются частично, по выбору эксперта. Корректность реализации процедуры означает, что реализация функционально соответствует СТБ 34.101.26 и что реализация не содержит ошибок и уязвимостей.

Эксперт должен проверить по крайней мере следующие аспекты реализации процедуры формирования запроса:

1 Для формирования значения компонента certID должен использоваться алгоритм хэширования, который определен в действующем ТНПА, исключая СТБ 1176.1.

2 Если запрос подписывается, то:

– реализация должна использовать для подписи личный ключ OCSP-клиента, запрашивающего информацию о статусе сертификата;

– реализация должна включать в запрос идентификатор криптографического алгоритма, который используется для подписи запроса;

– реализация должна включать в запрос информацию, которая определяет долговременные параметры алгоритмов, используемых для подписи запроса.

6.3.4 Корректность процедуры формирования запроса

При анализе корректности реализации процедуры разбора запроса исходные тексты программы оцениваются частично, по выбору эксперта. Корректность реализации процедуры означает, что реализация функционально соответствует СТБ 34.101.26 и что реализация не содержит ошибок и уязвимостей.

Эксперт должен проверить по крайней мере следующие аспекты реализации процедуры разбора запроса:

1 Для формирования значения компонента certID использовалься алгоритм хэширования, который определен в действующем ТНПА, исключая СТБ 1176.1;

2 Если запрос подписывается, то:

– реализация должна проверять, что ЭЦП запроса действительна;

– реализация должна проверять ЭЦП запроса алгоритмом с идентификатором, включенным в запрос;

– реализация должна проверять ЭЦП запроса алгоритмом с долговременными параметрами, информация о которых включена в запрос.

6.3.5 Корректность процедуры формирования запроса

При анализе корректности реализации процедуры формирования ответа исходные тексты программы оцениваются частично, по выбору эксперта. Корректность реализации процедуры отвечает, что реализация функционально соответствует СТБ 34.101.26 и что реализация не содержит ошибок и уязвимостей.

Эксперт должен проверить по крайней мере следующие аспекты реализации процедуры формирования ответа:

1 Реализация должна подписывать ответ на личном ключе стороны, которая для этого уполномочена (см. п. 5.3 СТБ 34.101.26).

2 Реализация должна включать в запрос идентификатор криптографического алгоритма, который используется для подписи запроса.

3 Реализация должна включать в запрос информацию, которая определяет долговременные параметры алгоритмов, используемые для подписи запроса.

6.3.6 Корректность процедуры формирования запроса

При анализе корректности реализации процедуры разбора ответа исходные тексты программы оцениваются частично, по выбору эксперта. Корректность реализации процедуры означает, что реализация функционально соответствует СТБ 34.101.26 и что реализация не содержит ошибок и уязвимостей.

Эксперт должен проверить по крайней мере следующие аспекты реализации процедуры разбора ответа:

1 Реализация должна проверять ЭЦП ответа на открытом ключе стороны, подписавшей ответ.

2 Реализация должна проверять ЭЦП ответа алгоритмом с идентификатором, включенным в ответ.

3 Реализация должна проверять ЭЦП ответа алгоритмом с долговременными параметрами, информация о которых включена в ответ.

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.

Эксперт проверяет корректность обработки исключительных ситуаций при выполнении проверок, проводимых в п. 6.3.1 – 6.3.6.

Для анализа корректности обработки исключительных ситуаций эксперт проверяет, что:

1 После каждого вызова функции, выполнение которой может приводить к возникновению исключительной ситуации, 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 согласно документации на вызываемую функцию.

3 Исключительные ситуации обрабатываются адекватно (возвращаются верные коды ошибок и сообщения об ошибках и т.п.).

6.3.8 Отсутствие недокументированных возможностей

Эксперт определяет отсутствие недокументированных возможностей по результатам проверок, выполненных в п. 6.3.1 – 6.3.7.

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
программы {Поле 4}, реализующей онлайновый протокол проверки статуса сертификата согласно СТБ 34.101.26-2012

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 онлайнового протокола проверки статуса сертификата согласно СТБ 34.101.26-2012.

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 онлайнового протокола проверки статуса сертификата согласно СТБ 34.101.26. 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
программы {Поле 4}, реализующей онлайновый протокол проверки статуса сертификата согласно СТБ 34.101.26-2012

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	CreateReqTest	{Поле 8}
2	ValidReqTest	{Поле 8}
3	InvalidReqTest	{Поле 8}
...	...	...

3. Заключение по результатам тестирования: программа {Поле 4} соответствует (не соответствует) требованиям, установленным в СТБ 34.101.26-2012.

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 34.101.26. 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
программы {Поле 4}, реализующей онлайновый протокол проверки статуса сертификата согласно СТБ 34.101.26-2012

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 криптографических алгоритмов	{Поле 8}
2	Корректность использования секретных параметров	{Поле 8}
3	Корректность уничтожения значений секретных параметров	{Поле 8}
4	Корректность процедуры формирования запроса	{Поле 8}
5	Корректность процедуры разбора запроса	{Поле 8}
6	Корректность процедуры формирования ответа	{Поле 8}
7	Корректность процедуры разбора ответа	{Поле 8}
8	Корректность обработки исключительных ситуаций	{Поле 8}
9	Отсутствие недокументированных возможностей	{Поле 8}

3. Заключение по результатам анализа исходных текстов программ: программа {Поле 4} соответствует требованиям, установленным в СТБ 34.101.26–2012.

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 34.101.26. 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.

Приложение Г
Тестовое программное обеспечение

Г.1 Программы преобразования АСН.1-файлов в текстовое представление

Программы, описанные в настоящем подразделе, является свободно распространяемыми программами, которые предназначены для анализа содержимого двоичных файлов, содержащих закодированные значения типов АСН.1. Они позволяют преобразовывать закодированные бинарные файлы в их текстовое представление.

Г.1.1 Программа `dumpasn1.exe`

Программа `dumpasn1.exe` является консольным приложением. Тексты программы располагаются по адресу: `https://www.cs.auckland.ac.nz/~pgut001/dumpasn1.c`.

Для преобразования файла запроса на получение сертификата в файл с текстовым представлением может использоваться, например, следующая команда:

```
dumpasn1.exe -t -a -z src.bin > dst.txt
```

В команде параметры имеют следующие значения: `src.bin` — закодированный исходный файл; `dst.txt` — текстовое представление исходного файла; `t` — отображение значений компонент в текстовом виде; `a` — отображение целиком блоков данных, длина которых больше 128 байтов; `z` — допущение полей нулевой длины.

Распространенные идентификаторы объектов могут передаваться в программу через конфигурационный файл `dumpasn1.cfg`. Стандартный конфигурационный файл периодически обновляется и размещается по адресу `https://www.cs.auckland.ac.nz/~pgut001/dumpasn1.cfg`. Для распознавания дополнительных идентификаторов, определенных в отечественных криптографических стандартах, может использоваться расширение конфигурационного файла, размещенное по адресу `https://github.com/agievich/bee2/blob/master/doc/dumpasn1by.cfg`.

Г.1.2 Программа `ASN.1 Editor`

Программа `ASN.1 Editor` является приложением операционной системы Windows с графическим пользовательским интерфейсом. Исполняемый файл программы располагается по адресу: `http://www.codeproject.com/Articles/4910/ASN-Editor`.

Для преобразования файла запроса на получение сертификата в текстовое представление необходимо открыть файл с помощью пункта основного меню программы: File → Open.

Г.2 Программы автоматической генерации тестовых наборов

Г.2.1 Технология Fuzzing

Для автоматизации тестирования программ, обрабатывающих сложные форматы данных, часто применяется технология Fuzzing. Специальная программа `fuzzer` обрабатывает испытываемую программу `prg`, которая в свою очередь обрабатывает файл `file`. Программа `fuzzer` выполняет многочисленные модификации `file`, подает эти модификации на вход `prg` и оценивает реакцию. Интерес для `fuzzer` представляют всевозможные

исключительные ситуации: зависания (hungs), утечки памяти (leaks), нарушение утверждений времени компиляции (asserts) и т. д. Любое из найденных исключений для испытуемой криптографической программы недопустимо.

Программа `fuzzer` выполняет модификации `file` разными способами. В большинстве случаев модификации формируются случайно, и тогда тестирование проходит по принципу «черный ящик». Намного больше ошибок можно выявить тогда, когда `fuzzer` учитывает (частично или полностью) формат `file`, т.е. поддерживает принцип «серый ящик» или даже «белый ящик».

Г.2.2 Программа AFL

Программа `AFL` (American Fuzzy Lop) — это свободно распространяемый `fuzzer`, с помощью которого найдено большое число уязвимостей в криптографических продуктах. Вся необходимая информация об `AFL`, в том числе документация и исходные файлы, размещена по адресу `http://lcamtuf.coredump.cx/afl`.

Испытуемая программа `prg` должна компилироваться средствами `AFL`, в свою очередь основанных на инструментах GCC или CLANG. Примерный make-файл для сборки испытуемой программы:

```
CC = path_to_afl/afl-gcc
CXX = path_to_afl/afl-g++
LDFLAGS = ...
CFLAGS = ...
PROGS = prg

all: $(PROGS)
prg: prg.c
  $(CC) $(CFLAGS) $@.c -o $@ $(LDFLAGS)
clean:
  rm -f $(PROGS) *.o ...
```

Испытуемая программа должна принимать на вход файл `file`. Тестовый файл, в окрестности которого будет организовано тестирование, помещается в специальный каталог `tests`. В это каталог могут быть добавлены любые другие тестовые файлы. Модификации `file`, которые привели к исключениям в `prg`, помещаются в каталог `findings`.

Тестирование запускается по команде

```
path_to_afl/afl-fuzz -i tests -o findings path_to_prg/prg @@
```

Информацию по дополнительным опциям команды, а также дополнительным возможностям `AFL` можно найти на упомянутом сайте.

Перед сборкой `AFL` в виртуальной среде на платформе Windows следует включить директиву `SIMPLE_FILES` в заголовочном файле `config.h` и в функции `trim_case()` модуля `afl_fuzz.c` строку

```
fd = open(q->fname, O_WRONLY | O_CREAT | O_EXCL, 0600);
```

изменить на

```
fd = open(q->fname, O_WRONLY | O_CREAT, 0600);
```

Приложение Д
Описание тестовых данных

В данном приложении приводится описание запросов и ответов онлайнового протокола проверки статуса сертификата (OCSP), используемых в тестах известного ответа. Каждый запрос/ответ в тестах основан на базовом запросе/ответе. Этот базовый запрос/ответ содержат типичные для всех запросов/ответов значения основных компонентов. При описании тестовых запросов/ответов приводятся лишь значения компонентов, которые отличаются от указанных в базовых запросах/ответах.

Д.1 Запросы на проверку статуса сертификата

Д.1.1 Запрос VReqTest1

Является базовым криптографическим сообщением.

Имеет следующие значения основных компонент:

Компонент сообщения или тип ASN.1	Значение компонента	Пояснения
`OCSPRequest`		Запрос на проверку статуса сертификата
`tbsRequest`		Информационная часть запроса
`requestList`		
`Request`		
`reqCert`		Информация о проверяемом сертификате
`hashAlgorithm`		
`algorithm`	1.2.112.0.2.0.34.101.31.81	Соответствует алгоритму `belt-hash`
`parameters`	`NULL`	Параметры не задаются
`issuerNameHash`	{Определенное значение}	Хэш-значение, вычисленное алгоритмом `belt-hash`
`issuerKeyHash`	{Определенное значение}	Хэш-значение, вычисленное алгоритмом `belt-hash`
`serialNumber`	3	Серийный номер сертификата, статус которого требуется проверить
`requestExtensions`		
`extnID`	1.3.6.1.5.5.7.48.1.2	Идентификатор расширения `ocspNonce` (см. п. 6.3 СТБ 34.101.26)
`extnValue`	{Определенное значение}	Значение из 16 октетов

Д.1.2 Криптографическое сообщение VReqTest2

Основано на базовом запросе.

Содержит компонент `optionalSignature`, содержащий подпись от информационной части запроса, и компонент `requesterName`, содержащий имя стороны, которая запрашивает информацию о статусе.

Д.1.3 Криптографическое сообщение VReqTest3

Основано на базовом запросе.

Компонента `requestList` содержит дополнительно еще одно значение типа `Request`.

Д.1.4 Криптографическое сообщение ISignReqTest4

Основано на запросе VReqTest2.

Содержит измененное значение компонента `optionalSignature.signature`.

Д.1.5 Криптографическое сообщение IVerReqTest5

Основано на базовом сообщении.

Значение компонента `version`: 9.

Д.1.6 Криптографическое сообщение IDigAlgIDReqTest6

Основано на базовом сообщении.

Значение компонента `tbsRequest.requestList.reqCert.hashAlgorithm.algorithm`: 1.2.112.0.2.0.34.101.31.51.

Д.1.7 Криптографическое сообщение INameHashReqTest7

Основано на базовом сообщении.

Значение компонента `tbsRequest.requestList.reqCert.issuerNameHash` имеет длину, которая не соответствует длине хэш-значения алгоритма `belt-hash`.

Д.1.8 Криптографическое сообщение IKeyHashReqTest8

Основано на базовом сообщении.

Значение компонента `tbsRequest.requestList.reqCert.issuerKeyHash` имеет длину, которая не соответствует длине хэш-значения алгоритма `belt-hash`.

Д.2 Ответы со статусом сертификата

Д.2.1 Ответ VRespTest1

Является базовым ответом.

Имеет следующие значения основных компонент:

Компонент сообщения или тип ASN.1	Значение компонента	Пояснения
`OCSPResponse`		Ответ со статусом сертификата
`responseStatus`	0	Соответствует значению «Ответ действителен»
`responseBytes`		
`responseType`	1.3.6.1.5.5.7.48.1.1	Соответствует ответу типа `ocspBasic`
`response`		
`tbsResponseData`		
`responderID`	{Определенное значение}	Значение типа `Name`
`producedAt`	{Определенное значение}	Содержит время создания ответа
`responses`		
`certID`		
`hashAlgorithm`		
`algorithm`	1.2.112.0.2.0.34.101.31.81	Соответствует алгоритму `belt-hash`
`parameters`	`NULL`	Параметры не задаются
`issuerNameHash`	{Определенное значение}	Хэш-значение, вычисленное алгоритмом `belt-hash`
`issuerKeyHash`	{Определенное значение}	Хэш-значение, вычисленное алгоритмом `belt-hash`
`serialNumber`	3	Серийный номер сертификата, статус которого требуется проверить
`certStatus`	`NULL`	Соответствует действующему сертификату
`thisUpdate`	{Определенное значение}	Содержит время создания ответа
`responseExtensions`		
`extnID`	1.3.6.1.5.5.7.48.1.2	Идентификатор расширения `ocspNonce` (см. п. 6.3 СТБ 34.101.26)
`extnValue`	{Определенное значение}	Значение из 16 октетов
`signatureAlgorithm`		
`algorithm`	1.2.112.0.2.0.34.101.45.12	Соответствует алгоритму `bign-with-hbelt`
`parameters`	`NULL`	
`signature`	{Определенное значение}	Содержит значение подписи
`certs`	{Определенное значение}	Содержит значение сертификата OCSP-сервера

Д.2.2 Криптографическое сообщение VRespTest2

Основано на базовом сообщении.

Содержит дополнительный элемент в компоненте `responses` для действующего сертификата.

Д.2.3 Криптографическое сообщение VRespTest3

Основано на базовом сообщении.
Содержит элемент в компоненте `responses` для отозванного сертификата.

Д.2.4 Криптографическое сообщение IStatusRespTest4

Основано на базовом сообщении.
Значение компонента `responseStatus`: 9.

Д.2.5 Криптографическое сообщение ISignRespTest5

Основано на базовом сообщении.
Значение компонента `signature` изменено.

Д.2.6 Криптографическое сообщение ITypeRespTest6

Основано на базовом сообщении.
Значение компонента `responseType`: 1.2.840.113549.1.7.1.

Д.2.7 Криптографическое сообщение ISignAlgIDRespTest7

Основано на базовом сообщении.
Значение компонента `signatureAlgorithm.algorithm`: 1.2.112.0.2.0.34.101.45.22.