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 34.101.65-2014

МИ.10165.10.01

Листов 23

Минск 2022

Предисловие

Настоящая методика испытаний предназначена для использования в испытательных лабораториях при проведении сертификационных испытаний средств криптографической защиты информации на соответствие требованиям СТБ 34.101.65-2014 «Информационные технологии и безопасность. Протокол защиты транспортного уровня (TLS)».

Содержание

1 Нормативные ссылки

В настоящем документе использованы ссылки на следующие стандарты:

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 34.101.65-2014 «Информационные технологии и безопасность. Протокол защиты транспортного уровня (TLS)».

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 34.101.65, а также следующие сокращения:

ЕСПД единая система программной документации;

TLS transport layer security (защита транспортного уровня).

3 Объект и цель испытаний

На испытания предоставляется программа, реализующая протокол TLS согласно СТБ 34.101.65. Целью испытаний является проверка соответствия реализации протокола TLS требованиям СТБ 34.101.65.

Целью испытаний является проверка соответствия протокола TLS, реализованного в объекте испытаний, требованиям СТБ 34.101.65.

4 Требования к объекту испытаний

Программа объекта испытаний должна реализовывать протокол TLS с криптонабором `TLS_DHE_BIGN_WITH_BELT_CTR_MAC_HBELT`. Программа может дополнительно реализовывать протокол TLS с одним или несколькими криптонаборами из следующего списка:

- `TLS_DHE_BIGN_WITH_BELT_DWP_HBELT`;
- `TLS_DHT_BIGN_WITH_BELT_CTR_MAC_HBELT`;
- `TLS_DHT_BIGN_WITH_BELT_DWP_HBELT`;
- `TLS_DHE_PSK_BIGN_WITH_BELT_CTR_MAC_HBELT`;
- `TLS_DHE_PSK_BIGN_WITH_BELT_DWP_HBELT`;
- `TLS_DHT_PSK_BIGN_WITH_BELT_CTR_MAC_HBELT`;
- `TLS_DHT_PSK_BIGN_WITH_BELT_DWP_HBELT`.

Программа должна реализовывать клиентскую часть TLS, серверную часть или обе части. Программа поддерживать, по крайней мере, один из уровней стойкости: 128, 192 или 256.

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

– в программе должен быть точно и полно реализован протокол TLS с заявленными криптонаборами;

– программа 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 34.101.65. В протоколе перечисляются реализованные криптонаборы протокола TLS, поддерживаемые стороны и уровни стойкости. 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 протокола или субпротоколов СТБ 34.101.65, которые в составе других программ имеют действующие сертификаты соответствия требованиям СТБ 34.101.65, то проверки по тестированию и анализу исходных текстов для данных реализаций могут не проводиться. При этом для подтверждения соответствия объекта испытаний требованиям СТБ 34.101.65 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 34.101.65 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 протоколе TLS, предварительно должна пройти успешные испытания по согласованной с Органом по сертификации методике испытаний.

Испытываемая программа может не поддерживать необязательный функционал, определенный в СТБ 34.101.65 (например, использование криптонаборов, реализующих механизм `PSK`). При этом сужение программой обязательного функционала, определенного в СТБ 34.101.65, 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 протокола TLS. Тестирование клиента (сервера) состоит в установке соединения с эталонным сервером (клиентом) и передачи данных по этому соединению. Тестирование завершено успешно, если соединение установлено и по нему корректно переданы данные.

Тестирование проводится либо локально, либо удаленно. При локальном тестировании испытуемый клиент и эталонный сервер либо испытуемый сервер и эталонный клиент размещаются на одной рабочей станции. При удаленном тестировании — на разных рабочих станциях, связанных сетью Интернет.

Тестирование проводится со следующими параметрами:

- тип тестирования: локальное, удаленное (через Интернет);
- тестируемая роль: сервер, клиент;
- используемый криптонабор;
- уровень стойкости: 128, 192, 256;

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.

5.4 Анализ исходных текстов

Эксперт проводит анализ исходных текстов путем проверки корректности реализации в испытуемой программе протокола TLS СТБ 34.101.65. Такой анализ состоит в получении экспертных заключений, касающихся:

- корректности использования криптографических алгоритмов (см. п. 6.4.1);
- корректности управления секретными данными (см. п. 6.4.2);

– корректности реализации субпротокола Record (см. п. 6.4.3);
– корректности реализации субпротокола Handshake (см. п. 6.4.4);
– корректности реализации субпротокола Change Cipher Spec (см. п. 6.4.5);
– корректности реализации субпротокола Alert (см. п. 6.4.6);
– корректности обработки исключительных ситуаций (см. п. 6.4.7);
– отсутствия недокументированных возможностей (см. п. 6.4.8).

При анализе исходных текстов проверки из вышеуказанных пунктов выполняются для всех реализованных криптонаборов и всех поддерживаемых сторон и уровней стойкости. При выполнении 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 протокол TLS, 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 протокол TLS;

– 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 протокол TLS;

– описание функций, реализующих протокол TLS, 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 протокола TLS 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

6.3 Общие сведения

При выполнении тестирования используются программы описанные в приложении Г. Для тестирования клиента соответственно используется только программа сервера (см. приложение Г.2). Для тестирования сервера используется как программа клиента (см. приложение Г.3) так и сервера.

6.3.1 Локальное тестирование протокола TLS

Локальное тестирование клиента проводится каждого набора параметров `x.y.z` (`x` — тестируемая роль, `y` — криптонабор, `z` — уровень стойкости) в соответствии с таблицей 1.

Если `x = клиент`, то выполняется следующий тест:

Тест Local.client.y.z

1 Сгенерировать ключ `priv.key` и сертификат `cert.pem` сервера используя кривую Bign с уровнем стойкости `z` следующим способом:

```
$ openssl genpkey -algorithm bign -pkeyopt params:bign-curve
   <2*z> -out priv.key
$ openssl req -x509 -subj "<subj>" -new -key priv.key -nodes
   -out cert.pem
```

2 Запустить сервер по адресу `host:port` (например `127.0.0.1:443`) при использовании криптонабором `y` механизма `DHE_BIGN`, `DHT_BIGN`или `DHT_PSK_BIGN`, следующей командой:

```
$ openssl s_server -accept <host:port> -cipher <y> \
    -cert cert.pem -key priv.key -tls1_2
```

По необходимости указываются дополнительные параметры `-psk`, `-psk_identity` и `-psk_hint`. Если криптонабором `y` используется механизм `DHE_PSK_BIGN` сервер запускается командой:

```
$ openssl s_server -accept <host:port> -tls1_2 \
    -psk <psk> -psk_hint <hint>
```

Вохможные значения параметров:
- <psk> — секрет, любая строка в шестнадцатеричном виде (1f2f3f);
- <hint> — номер секрета.

3 Тестируемым клиентом подключиться к серверу по адресу `host:port`.
4 Возвратить `УСПЕХ`, если соединение установлено и передача данных прошла успешно.
5 Возвратить `ОШИБКА`.

Если `x = сервер`, то выполняется следующий тест:

Тест Local.server.y.z

1 Подключиться к тестируемому серверу расположенному по адресу `host:port` используя кривую `curve` уровня стойкости `z` следующей командой:

```
$ openssl s_client -connect <host:port> -cipher <y> \
    -curves bign-curves<2*z> -tls1_2
```

По необходимости указываются дополнительные параметры `-psk`, `-psk_identity`.
2 Возвратить `УСПЕХ`, если соединение установлено и передача данных прошла успешно.
3 Возвратить `ОШИБКА`.

Таблица 1: Локальное тестирование

Роль	Криптонабор	Уровень стойкости
Клиент	TLS_DHE_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHE_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256
	TLS_DHT_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHT_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256
	TLS_DHE_PSK_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHE_PSK_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256
	TLS_DHT_PSK_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHT_PSK_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256
Сервер	TLS_DHE_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHE_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256
	TLS_DHT_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHT_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256
	TLS_DHE_PSK_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHE_PSK_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256
	TLS_DHT_PSK_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHT_PSK_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256

6.3.2 Удаленное тестирование протокола TLS

Удаленное тестирование клиента проводится для каждого набора параметров `x.y.z` (`x` — тестируемая роль, `y` — криптонабор, `z` — уровень стойкости) в соответствии с таблицей 2.

Для тестирования клиента потребуется браузер, поддерживающий криптонаборы СТБ 34.101.65. Браузер Epiphany может использоваться для такой поддержки. Процесс его сборки, только для эталонного клиента описан в приложении Г.3.

Если `x = клиент`, то выполняется следующий тест:

Тест Remote.client.y.z

1 Открыть браузер на тестируемом клиенте, поддерживающем криптонаборы СТБ 34.101.65.
2 Запустить эталонный сервер в соответствии с приложением Г.2.
3 В адресную строку браузера ввести адрес эталонного сервера с портом равным:
– 8443, если <y> = `TLS_DHE_BIGN_WITH_BELT_CTR_MAC_HBELT`;
– 8444, если <y> = `TLS_DHE_BIGN_WITH_BELT_DWP_HBELT`;
– 8445, если <y> = `TLS_DHT_BIGN_WITH_BELT_CTR_MAC_HBELT`;
– 8446, если <y> = `TLS_DHT_BIGN_WITH_BELT_DWP_HBELT`.
4 Возвратить `УСПЕХ`, если в поле `Current Ciphersuite` отображен криптонабор `y`.
5 Возвратить `ОШИБКА`.

Если `x = сервер`, то выполняется следующий тест:

Тест Remote.server.y.z

1 Собрать эталонного клиента в соответствии с приложением Г.3.
2 Открыть браузер эталонного клиента.
3 Запустить эталонный сервер в соответствии с приложением Г.2.
4 В адресную строку браузера ввести адрес эталонного сервера с портом равным `443`.
5 В текстовое поле полученной веб страницы ввести адрес тестируемого сервера, использующего кривую стойкости <z>, и нажать кнопку `Test` страницы.
6 Для каждого криптонабора будет отображена информация в виде: зеленой галочки или красного креста, сигнализирующих соответственно о поддержке данного криптонабора. Возвратить `УСПЕХ` если напротив криптонабора `y` отображена зеленая галочка.
7 Возвратить `ОШИБКА`.

Таблица 2: Удаленное тестирование

Роль	Криптонабор	Уровень стойкости
Клиент	TLS_DHE_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHE_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256
	TLS_DHT_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHT_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256
Сервер	TLS_DHE_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHE_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256
	TLS_DHT_BIGN_WITH_BELT_CTR_MAC_HBELT	128, 192, 256
	TLS_DHT_BIGN_WITH_BELT_DWP_HBELT	128, 192, 256

6.4 Анализ исходных текстов

6.4.1 Корректность использования криптографических алгоритмов

В программе используются алгоритмы шифрования, электронной цифровой подписи, имитозащиты, генерации псевдослучайных чисел, формирования общего ключа.

Использование функций, реализующих криптографический алгоритм, должно выполняться в соответствии с СТБ 34.101.65, 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.

6.4.2 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 реализации TLS используются следующие секретные данные:

– предварительный мастер-ключ;

– мастер ключ;

– ключ имитозащиты клиент;
– ключ имитозащиты сервер;
– ключ шифрования клиента;
– ключ шифрования сервера;
– личный ключ ЭЦП сервера.

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.

6.4.3 Корректность реализации субпротокола Record

Протокол Record обеспечивает конфиденциальность и контроль целостности транспортируемых данных. Каждый фрагмент данных при передаче протоколом Record дополняется полями, которые указывают на тип содержимого и длину фрагмента.

Реализация субпротокола Record не должна отправлять данные, если тип их содержимого отличен от одного из следующих: `change_cipher_spec`, `alert`, `handshake`, `application_data`. При получении данных неопределенного типа должно высылаться криптическое сигнальное сообщение `unexpected_message`.

Протокол Record обрабатывает отправляемые данные с использованием текущего состояния записи, а принимаемые — с использованием текущего состояния чтения. Состояние — это структура данных, которая определяет используемые алгоритмы сжатия, шифрования и имитозащиты, а также состояния и параметры этих алгоритмов. Состояние может быть текущим или ожидаемым. Эксперт проверяет, что порядковый номер `seq_num` сбрасывается в 0 всякий раз, когда состояние соединения становится текущим, и увеличиваться на единицу после обработки (передачи или приема) каждого фрагмента данных. Порядковый номер является числом и не может превышать значения $2^{64} - 1$. Порядковые номера не должны повторяться. Поэтому при достижении максимального порядкового номера реализации TLS должны переустанавливать соединение. Также при использовании алгоритма шифрования `belt-ctr` в качестве синхропосылки используется 8 байтовый порядковый номер дополненный 8 нулевыми байтами, а при использовании алгоритма шифрования `belt-dwp` все явные части синхропосылок различаются.

6.4.4 Корректность реализации субпротокола Handshake

Протокол Handshake выполняется поверх протокола Record и отвечает за установку, возобновление или переустановку связи. Протокол Handshake позволяет клиенту и серверу аутентифицировать друг друга, а также согласовать криптографические алгоритмы и общие ключи до того, как прикладной протокол начнет прием или передачу данных.

Эксперт должен проверить, по крайне мере, следующие аспекты реализации субпротокола Handshake:

– при использовании алгоритма формирования общего ключа `DHE_BIGN` сервер в сообщении `ServerKeyExchange` передает эфемерный открытый ключ;

– при использовании алгоритма формирования общего ключа `DHT_BIGN` и `DHT_PSK_BIGN` в процессе создания и обработки токена ключа используется нулевой заголовок ключа;

– вырабатывается ключ шифрования длиной 256 бит;

– вырабатывается ключ имитозащиты длиной 256 бит;

– при использовании алгоритма формирования общего ключа `DHE_PSK_BIGN` сервер в сообщении `ServerKeyExchange` передает идентификатор эллиптической кривой, причем должны использоваться идентификаторы заданные в СТБ 34.101.45 (приложение Д). Если при определении параметров или при проверке открытого ключа произошла ошибка, то клиент должен прекратить Handshake;

– при использовании алгоритма формирования общего ключа `DHE_PSK_BIGN` предварительный масте-ключ должен быть сформирован следующим образом: `len(other_secret) + other_secret + len(psk) + psk`;

– при использовании алгоритма формирования общего ключа `DHT_PSK_BIGN` сервер пересылает клиенту свой сертификат в сообщении `Certificate`.

6.4.5 Корректность реализации субпротокола Change Cipher Spec

Субпротокол Change Cipher Spec сообщает о смене параметров защиты на новые, согласованные при выполнении субпротокола Handshake. В этом протоколе используется единственное сообщение `ChangeCipherSpec`, которое защищается и сжимается с использованием текущего состояния соединения. Таким образом эксперт проверяет, что после отправки сообщения `ChangeCipherSpec` каждая из сторон поменяет старые параметры на новые.

6.4.6 Корректность реализации субпротокола Alert

Субпротокол Record поддерживает передачу сигнальных сообщений. Эти сообщения формируются субпротоколом Alert и передают уровень сигнала (предупреждение или критическая ошибка) и его описание. При передаче или получении критического сообщения стороны не должны использовать идентификаторы сеанса и ключи связанные с соединением, закрытым из-за ошибки.

Эксперт проверяет что сигнальные сообщения передаются:

– при проверке клиентом сертификата и ЭЦП сервера произошла ошибка;

– при проверке сервером полученного от клиента эфемерного открытого ключа произошла ошибка;

– при разборе сервером токена транспортируемого ключа;

– при определении параметров эллиптической кривой произошла ошибка;

– при получении сообщения `close_notify`;

– при получении некорректного сообщения.

6.4.7 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. 6.4.1 – 6.4.6.

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.).

6.4.8 Отсутствие недокументированных возможностей

Эксперт определяет отсутствие недокументированных возможностей по результатам проверок, выполненных в п. 6.4.1 – 6.4.7.

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 протокол TLS согласно СТБ 34.101.65-2014

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 протокола согласно СТБ 34.101.65-2014.

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

В поле 1 указывается номер экземпляра протокола.

В поле 2 указывается номер, однозначно идентифицирующий протокол.

В поле 3 указывается дата составления протокола.

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

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

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

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

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

В поле 13 указывается результат выполнения проверки: «положительно» — результат проверки положительный, «отрицательно» — результат проверки отрицательный. После завершения анализа документации и заполнения таблицы делается вывод о соответствии (не соответствии) документации программе объекта испытаний в части реализации протокола согласно СТБ 34.101.65. 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 протокол TLS согласно СТБ 34.101.65-2014

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	Local.client.DHE-BIGN-WITH-BELT-CTR-MAC-HBELT.128	{Поле 8}
2	Local.server.DHE-BIGN-WITH-BELT-CTR-MAC-HBELT.128	{Поле 8}
3	Remote.client.DHE-BIGN-WITH-BELT-CTR-MAC-HBELT.128	{Поле 8}
4	Remote.server.DHE-BIGN-WITH-BELT-CTR-MAC-HBELT.128	{Поле 8}
...	...	...

3. Заключение по результатам тестирования: программа {Поле 4} соответствует (не соответствует) требованиям, установленным в СТБ 34.101.65-2014.

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 34.101.65. 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 протокол TLS согласно СТБ 34.101.65-2014

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 реализации вложенного в TLS субпротокола Record	{Поле 8}
5	Корректность реализации вложенного в TLS субпротокола Handshake	{Поле 8}
6	Корректность реализации вложенного в TLS субпротокола Change Cipher Spec	{Поле 8}
7	Корректность реализации вложенного в TLS субпротокола Alert	{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 34.101.65–2014.

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 34.101.65. 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 Локальное тестирование

При локальном тестировании используется библиотека OpenSSL вместе с эталонной реализацией TLS с криптонаборами СТБ 34.101.65. Исходные тексты эталонной реализации находятся по адресу: `https://github.com/bcrypto/bee2evp/`.

Сборка реализации происходит следующим образом:

```
$ cd doc
$ bash build.sh
```

После этого будет собрана библиотека OpenSSL, поддерживающая криптонаборы СТБ 34.101.65. Для успешного вызова команд OpenSSL необходимо добавить в переменные PATH и LD_LIBRARY_PATH пути до бинарного файла `openssl` и директории `lib` соответственно. Бинарный файл после сборки будет находиться в директории `bee2evp/build/local/bin`, а директория `lib` — в `bee2evp/build/local`.

Для запуска локального сервера выполняется команда `openssl s_server` с нужными параметрами, а для запуска клиента — команда `openssl s_client`.

Г.2 Программы сервера

Программы, описанные далее, являются свободно распространяемыми контейнерами, которые предназначены для развертывания сервера поддерживающего белорусские криптонаборы.

Контейнеры собираются и развертываются с помощью приложений `docker` (`https://docs.docker.com/engine/install/ubuntu/`) и `docker-compose` (`https://docs.docker.com/compose/install/`). Программное обеспечение Docker — это платформа, которая предназначена для разработки, развёртывания и запуска приложений в контейнерах. Можно создавать целые системы и легко запускать их в любом месте, поддерживающем такой контейнер. Можно представить, что контейнеры подобны виртуальным машинам, по крайне мере с точки зрения обычного пользователя, но на самом деле это не виртуальная машина. Различия носят технический характер. Ключевое преимущество Docker состоит в том, что он позволяет пользователям упаковать приложение со всеми его зависимостями в стандартизированный модуль для разработки.

Для запуска контейнеров требуется скачать репозиторий `btls` (`https://github.com/bcrypto/btls/tree/master/server`).

Г.2.1 Программа `btls256`

Программа `btls256` является контейнерным приложением, реализующим сервер, использующий кривую `bign-curve256v1`. Исходные тексты программы располагаются по адресу: `https://github.com/bcrypto/btls/tree/master/server/btls256`.

Для запуска контейнера используются следующие команды:

```
$ sudo docker pull btls/btls256
$ sudo docker-compose up -d btls256
$ sudo docker exec -it btls256 bash
$ nginx -g "daemon off;"
```

Первая команда скачивает контейнер `btls256`, вторая запускает контейнер, с помощью третьей заходим внутрь контейнера и четвертая запускает сам сервер. После этого сервер готов принимать запросы от клиента.

Г.2.2 Программа `btls384`

Программа `btls384` является контейнерным приложением, реализующим сервер, использующий кривую `bign-curve384v1`. Исходные тексты программы располагаются по адресу: `https://github.com/bcrypto/btls/tree/master/server/btls384`.

Для запуска контейнера используются следующие команды:

```
$ sudo docker pull btls/btls384
$ sudo docker-compose up -d btls384
$ sudo docker exec -it btls384 bash
$ nginx -g "daemon off;"
```

Первая команда скачивает контейнер `btls386`, вторая запускает контейнер, с помощью третьей заходим внутрь контейнера и четвертая запускает сам сервер. После этого сервер готов принимать запросы от клиента.

Г.2.3 Программа `btls512`

Программа `btls512` является контейнерным приложением, реализующим сервер, использующий кривую `bign-curve512v1`. Исходные тексты программы располагаются по адресу: `https://github.com/bcrypto/btls/tree/master/server/btls512`.

Для запуска контейнера используются следующие команды:

```
$ sudo docker pull btls/btls512
$ sudo docker-compose up -d btls512
$ sudo docker exec -it btls512 bash
$ nginx -g "daemon off;"
```

Первая команда скачивает контейнер `btls512`, вторая запускает контейнер, с помощью третьей заходим внутрь контейнера и четвертая запускает сам сервер. После этого сервер готов принимать запросы от клиента.

Г.2.4 Программа `flask`

Программа `flask` является контейнерным приложением. Используется для тестирования удаленного сервера. Исходные тексты программы располагаются по адресу: `https://github.com/bcrypto/btls/tree/master/server/flask`.

Для запуска контейнера используются следующие команды:

```
$ sudo docker pull btls/flask
$ sudo docker-compose up -d flask
$ sudo docker exec -it flask bash
$ flask --host=0.0.0.0 --port=5000;
```

Первая команда скачивает контейнер `flask`, вторая запускает контейнер, с помощью третьей заходим внутрь контейнера и четвертая запускает сам сервер. После этого сервер готов принимать запросы от клиента.

Г.3 Программа клиента

Программа представленная в данном подразделе использует браузер Epiphany (`https://gitlab.gnome.org/GNOME/epiphany`), который для создания соединения использует библиотеку glib-networking (`https://gitlab.gnome.org/GNOME/glib-networking`), которая в свою очередь использует библиотеку OpenSSL, собранную с белорусскими криптонаборами. Исходные тексты программы располагаются по адресу: `https://github.com/bcrypto/btls/tree/master/client`.

Для сборки и запуска клиента выполняются следующие этапы:

1) скачать репозиторий `btls`;

2) перейти в директорию `client`;

3) запустить сборку клиента командной `bash build_client.sh`;

4) изменить переменную окружения PATH, LD_LIBRARY_PATH, и другие переменные, которые указаны в `add_to_bashrc.sh`. Это нужно для успешного запуска Epiphany и OpenSSL. Все изменения делаются вручную или запускается команда `bash add_to_bashrc.sh`, которая сама изменяет данные переменные;

5) запустить браузер Epiphany.