ФЕДЕРАЛЬНОЕ АГЕНТСТВО
ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

РЕКОМЕНДАЦИИ ПО СТАНДАРТИЗАЦИИ

Р 50.1.113
—
2016

Информационная технология

КРИПТОГРАФИЧЕСКАЯ ЗАЩИТА ИНФОРМАЦИИ

Криптографические алгоритмы, сопутствующие применению алгоритмов электронной цифровой подписи и функции хэширования

Издание официальное

Москва
Стандартинформ
2016

Предисловие

1 РАЗРАБОТАНЫ подкомитетом № 1 Технического комитета по стандартизации ТК 26 «Криптографическая защита информации»

2 ВНЕСЕНЫ Техническим комитетом по стандартизации ТК 26 «Криптографическая защита информации»

3 УТВЕРЖДЕНЫ И ВВЕДЕНЫ В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 28.11.2016 г. № 1828-ст

4 ВВЕДЕНЫ ВПЕРВЫЕ

Правила применения настоящих рекомендаций установлены в статье 26 Федерального закона «О стандартизации в Российской Федерации». Информация об изменениях к настоящим рекомендациям публикуется в ежегодном (по состоянию на 1 января текущего года) информационном указателе «Национальные стандарты», а официальный текст изменений и поправок – в ежемесячном информационном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящих рекомендаций соответствующее уведомление будет опубликовано в ближайшем выпуске ежемесячного информационного указателя «Национальные стандарты». Соответствующая информация, уведомление и тексты размещаются также в информационной системе общего пользования – на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет (www.gost.ru)

Содержание

Введение

Использование национальных криптографических алгоритмов, определенных ГОСТ Р 34.10 и ГОСТ Р 34.11, в средствах защиты информации осуществляют, как правило, в рамках криптографических протоколов, базирующихся на сопутствующих алгоритмах.

Настоящие рекомендации содержат описания сопутствующих алгоритмов, предназначенных для определения псевдослучайных функций протоколов, функций преобразования ключей, согласования ключей по протоколу Диффи-Хеллмана и экспорта ключевого материала.

Необходимость разработки настоящих рекомендаций вызвана потребностью в обеспечении совместимости криптографических протоколов различных производителей, использующих алгоритмы ГОСТ Р 34.10 и ГОСТ Р 34.11.

Примечание – Основная часть настоящих рекомендаций дополнена приложением А.

.

РЕКОМЕНДАЦИИ ПО СТАНДАРТИЗАЦИИ

Информационная технология

КРИПТОГРАФИЧЕСКАЯ ЗАЩИТА ИНФОРМАЦИИ

Криптографические алгоритмы, сопутствующие применению алгоритмов электронной цифровой подписи и функции хэширования

Дата введения — 2017—06—01

1 Область применения

Настоящие рекомендации предназначены для применения в информационных системах, использующих механизмы шифрования и защиты аутентичности данных, с использованием алгоритмов электронной (цифровой) подписи по ГОСТ Р 34.10 и функции хэширования по ГОСТ Р 34.11 в общедоступных и корпоративных сетях для защиты информации, не содержащей сведений, составляющих государственную тайну.

2 Нормативные ссылки

В настоящих рекомендациях использованы нормативные ссылки на следующие стандарты:

ГОСТ Р 34.10–2012 Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи

ГОСТ Р 34.11–2012 Информационная технология. Криптографическая защита информации. Функция хэширования

П р и м е ч а н и е – При пользовании настоящими рекомендациями целесообразно проверить действие ссылочных стандартов (рекомендаций) в информационной системе общего пользования – на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет или по ежегодному информационному указателю «Национальные стандарты», который опубликован по состоянию на 1 января текущего года, и по выпускам ежемесячного информационного указателя «Национальные стандарты» за текущий год. Если заменен ссылочный стандарт (рекомендации), на который дана недатированная ссылка, то рекомендуется использовать действующую версию этого стандарта

Издание официальное

(рекомендаций) с учетом всех внесенных в данную версию изменений. Если заменен ссылочный стандарт (рекомендации), на который дана датированная ссылка, то рекомендуется использовать версию этого стандарта (рекомендаций) с указанным выше годом утверждения (принятия). Если после утверждения настоящих рекомендаций в ссылочный стандарт (рекомендации), на который дана датированная ссылка, внесено изменение, затрагивающее положение, на которое дана ссылка, то это положение рекомендуется применять без учета данного изменения. Если ссылочный стандарт (рекомендации) отменен без замены, то положение, в котором дана ссылка на него, применяется в части, не затрагивающей эту ссылку.

3 Термины, определения и обозначения

3.1 Термины и определения

В настоящих рекомендациях применены следующие термины с соответствующими определениями:

3.1.1 **код аутентификации сообщения на основе хэш-функции HMAC**: Механизм обеспечения аутентичности информации на основе симметричного ключа, построенный с использованием хэш-функции.

3.1.2 **ключ K**: Произвольный элемент из V_n, если $K \in V_n$, то его длина равна n бит, где n может быть произвольным натуральным числом.

3.1.3 **псевдослучайная функция PRF**: Отображение, позволяющее вырабатывать псевдослучайные последовательности байтов.

3.1.4 **функция диверсификации KDF**: Отображение, позволяющее порождать производные ключи и ключевой материал по корневому ключу и произвольным данным с использованием псевдослучайной функции.

3.1.5 **выработка ключей обмена VKO**: Алгоритм согласования ключей (основан на алгоритме Диффи-Хеллмана и некоторой хэш-функции).

3.1.6 **битовое представление**: Для элемента $W = (w^0, w^1, \dots, w^{r-1})$, $W \in V_8^r$, битовым представлением называют вектор $(w_{8r-1}, w_{8r-2}, \dots, w_1, w_0) \in V_{8r}$, где $w^0 = (w_7, w_6, \dots, w_0), w^1 = (w_{15}, w_{14}, \dots, w_8), \dots, w^{r-1} = (w_{8r-1}, w_{8r-2}, \dots, w_{8r-8})$ являются элементами пространства V_8.

3.1.7 **байтовое представление**: Если n кратно 8, $r = n/8$, то байтовым представлением элемента $W = (w_{n-1}, w_{n-2}, \dots, w_0)$, $W \in V_n$, называют байтовый вектор $(w^0, w^1, \dots, w^{r-1}) \in V_8^r$, где $w^0 = (w_7, w_6, \dots, w_0), w^1 = (w_{15}, w_{14}, \dots, w_8), \dots, w^{r-1} = (w_{8r-1}, w_{8r-2}, \dots, w_{8r-8})$ являются элементами пространства V_8.

3.2 Обозначения

В настоящих рекомендациях использованы следующие обозначения:

$\oplus$ — операция покомпонентного сложения по модулю 2 двух двоичных строк одинаковой длины;

V_n — конечномерное векторное пространство над $GF(2)$ размерности n с операцией сложения $\oplus$; при n равном 0 пространство V_0 состоит из единственного пустого элемента длины 0; если U – элемент V_n, то в битовом представлении $U = (u_{n-1}, u_{n-2}, \dots, u_1, u_0)$, $u_i \in \{0,1\}$;

V_8^r — множество байтовых векторов длины r, $r \geq 0$; при $n = 0$ пространство V_0 состоит из единственного пустого элемента длины 0; если W– элемент V_8^r, $r > 0$, то в байтовом представлении $W = (w^0, w^1, \dots, w^{r-1})$, где $w^0, w^1, \dots, w^{r-1}$принадлежат пространству V_8;

$A|B$ — конкатенация байтовых векторов A и B; если $A = (a^0, a^1, \dots, a^{r1-1})$, $A \in V_8^{r1}$, $B = (b^0, b^1, \dots, b^{r2-1})$, $B \in V_8^{r2}$, то $A|B = (a^0, a^1, \dots, a^{r1-1}, b^0, b^1, \dots, b^{r2-1})$, $A|B \in V_8^{r1+r2}$;

H_{256} — хэш-функция H с длиной выхода, равной 256 битам, определенная в ГОСТ Р 34.11–2012 (раздел 8);

H_{512} — хэш-функция H с длиной выхода, равной 256 битам, определенная в ГОСТ Р 34.11–2012 (раздел 8).

4 Описание алгоритмов

Возможные значения аргументов функций в представленных алгоритмах ограничиваются допустимостью их использования в качестве входных параметров преобразований и присваиваются в протоколах.

Для выработки байтовой последовательности длины r с помощью функций, вырабатывающих последовательности большей длины, необходимо взять из выходной последовательности r первых байтов. Это замечание относится к следующим функциям, описанным в настоящих рекомендациях:

- функции, описанные в 4.2;
- алгоритм диверсификации KDF_TREE_GOSTR3411_2012_256, описанный в 4.5.

Далее все данные (элементы пространства V_n), если явно не указано иное, считают приведенными в байтовом представлении.

Если используемая функция определена вне настоящего документа (например, H_{256}) и ее определение использует аргументы в битовом представлении, то подразумевается, что битовое представление аргументов формируется непосредственно перед вычислением функции (в частности, только после применения операции «|» к байтовому представлению аргументов).

Если в качестве аргумента определяемых ниже функций использовано выходное значение другой функции, которая определена вне настоящих рекомендаций и имеет

выходные значение в битовом представлении, то подразумевается, что выходное значение перед подстановкой в аргументы переведено в байтовое представление.

4.1 Алгоритмы HMAC

В настоящем подразделе определены алгоритмы вычисления кода аутентификации сообщения HMAC на основе хэш-функции H, определенной в ГОСТ Р 34.11–2012 (раздел 8) с различными длинами выходных значений.

4.1.1 HMAC_GOSTR3411_2012_256

Алгоритм HMAC_GOSTR3411_2012_256 предназначен для вычисления кода аутентификации сообщения HMAC на основе хэш-функции H с длиной выхода, равной 256 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8), и имеет следующий идентификатор:

id-tc26-HMAC-gost-3411-12-256, «1.2.643.7.1.1.4.1».

Результатом работы данного алгоритма является значение функции $HMAC_{256}(K,T)$, вычисление которой для данных T произвольной длины на ключе K длины n бит состоит в формировании байтовой строки K^* длины 64 байта и выполнении преобразований над K^* и данными T с использованием хэш-функции H_{256}.

Допускаются любые значения длины n из интервала от 256 до 512.

Для формирования ключа K^* при $n < 512$ следует положить строку K^* равной байтовому представлению битовой строки $K|A$, где $A = (0, 0, \dots, 0) \in V_{512-n}$; если $n = 512$, положить K^* равной байтовому представлению K. Значение $HMAC_{256}(K,T)$ определено выражением:

$$HMAC_{256}(K,T) = H_{256}\left(K^* \oplus opad \mid H_{256}(K^* \oplus ipad \mid T)\right), \quad (1)$$

где в байтовом представлении

$$ipad = (0x36 \mid 0x36 \mid \dots \mid 0x36) \in V_8^{64},$$
$$opad = (0x5C \mid 0x5C \mid \dots \mid 0x5C) \in V_8^{64}.$$

Данный алгоритм использует H_{256} в качестве хэш-функции в конструкции HMAC, описанной в [1]. Указанный способ формирования $ipad$ и $opad$ также приведен в [1]. Длина выхода $HMAC_{256}$ равна 32 байтам, длина блока итерационной процедуры функции сжатия для H_{256} равна 64 байтам (в обозначениях [1]: $L = 32$, $B = 64$).

4.1.2 HMAC_GOSTR3411_2012_512

Алгоритм HMAC_GOSTR3411_2012_512 предназначен для вычисления кода аутентификации сообщения HMAC на основе хэш-функции H с длиной выхода, равной

512 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8), и имеет следующий идентификатор:

*id-tc26-*HMAC*-gost-3411-12-512 , «1.2.643.7.1.1.4.2».*

Результатом работы данного алгоритма является значение функции $HMAC_{512}(K,T)$, вычисление которой для данных T произвольной длины на ключе K длины n бит состоит в формировании байтовой строки K^* длины 64 байта и выполнении преобразований над K^* и данными T с использованием хэш-функции H_{512}.

Допускаются любые значения длины n из интервала от 256 до 512. Рекомендуется использовать значение n, равное 512.

Для формирования ключа K^* при $n < 512$ следует положить строку K^* равной байтовому представлению битовой строки $K|A$, где $A = (0, 0, \dots, 0) \in V_{512-n}$; если $n = 512$, положить K^* равной байтовому представлению K. Значение $HMAC_{512}\,(K,T)$ определено выражением:

$$HMAC_{512}(K,T) = H_{512}\left(K^* \oplus opad \mid H_{512}(K^* \oplus ipad \mid T)\right), \quad (2)$$

где в байтовом представлении

$$ipad = (0x36 \mid 0x36 \mid \dots \mid 0x36) \in V_8^{64},$$
$$opad = (0x5C \mid 0x5C \mid \dots \mid 0x5C) \in V_8^{64}.$$

Данный алгоритм использует H_{512} в качестве хэш-функции в конструкции HMAC, описанной в [1]. Указанный способ формирования $ipad$ и $opad$ приведен также в [1]. Длина выхода $HMAC_{512}$ равна 64 байтам, длина блока итерационной процедуры функции сжатия для H_{512} равна 64 байтам (в обозначениях [1]: $L = 64$, $B = 64$).

4.2 Псевдослучайные функции

В настоящем подразделе определены шесть рекомендуемых к использованию преобразований PRF – два для протокола TLS и четыре для протокола IPsec, построенных на основе HMAC.

4.2.1 Псевдослучайные функции протокола TLS

4.2.1.1 PRF_TLS_GOSTR3411_2012_256

Преобразование PRF_TLS_GOSTR3411_2012_256 на основе хэш-функции H с длиной выхода, равной 256 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8), задает псевдослучайную функцию протокола TLS (версии 1.0 и выше), использующую алгоритм HMAC_GOSTR3411_2012_256, описанный в 4.1.1. Результатом работы данного преобразования является значение функции PRF_{TLS_256}, аргументами которой являются байтовые строки $secret$, $label$, $seed$.

$$PRF_{TLS_{256}}(secret, label, seed) = P_{256}(secret, label \mid seed); \quad (3)$$

$$P_{256}(secret, S) = HMAC_{256}(secret, A_1 \mid S) \mid HMAC_{256}(secret, A_2 \mid S) \mid \ldots, \quad (4)$$

где параметры A_i определяют последовательно следующим образом:

$$A_0 = S,$$
$$A_i = HMAC_{256}(secret, A_{i-1}), \quad (5)$$

где индекс i изменяется в пределах, обеспечивающих выработку последовательности требуемой длины, определяемой протоколом.

Значения параметров $label$ и $seed$ должны задаваться протоколом, их длины должны быть фиксированы протоколом.

Функция P_{256} использует функцию $HMAC_{256}$, описанную в 4.1.1, и соответствует способу задания аргументов и выходного значения функции расширения данных *P_hash*, приведенному в разделе 5 [2] и использованному позднее в [5].

4.2.1.2 PRF_TLS_GOSTR3411_2012_512

Преобразование PRF_TLS_GOSTR3411_2012_512 на основе хэш-функции H с длиной выхода, равной 512 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8), задает псевдослучайную функцию протокола TLS (версии 1.0 и выше), использующую алгоритм HMAC_GOSTR3411_2012_512, описанный в 4.1.2. Результатом работы данного преобразования является значение функции PRF_{TLS_512}, аргументами которой являются байтовые строки $secret$, $label$, $seed$.

$$PRF_{TLS\_512}(secret, label, seed) = P_{512}(secret, label \mid seed); \quad (6)$$

$$P_{512}(secret, S) = HMAC_{512}(secret, A_1 \mid S) \mid HMAC_{512}(secret, A_2 \mid S) \mid \ldots, \quad (7)$$

где параметры A_i определяют последовательно следующим образом:

$$A_0 = S,$$
$$A_i = HMAC_{512}(secret, A_{i-1}), \quad (8)$$

где индекс i изменяется в пределах, обеспечивающих выработку последовательности требуемой длины, определяемой протоколом.

Значения параметров $label$ и $seed$ должны задаваться протоколом, их длины должны быть фиксированы протоколом.

Функция P_{512} использует функцию $HMAC_{512}$, описанную в 4.1.2, и соответствует способу задания аргументов и выходного значения функции расширения данных *P_hash*, приведенному в разделе 5 [2] и использованному позднее в [5].

4.2.2 Псевдослучайные функции протокола IPsec на основе ГОСТ Р 34.11, 256 бит

4.2.2.1 PRF_IPSEC_KEYMAT_GOSTR3411_2012_256

Преобразование PRF_IPSEC_KEYMAT_GOSTR3411_2012_256 на основе хэш-функции H с длиной выхода, равной 256 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8), задает псевдослучайную функцию выработки ключевого материала протокола IPsec, использующую алгоритм HMAC_GOSTR3411_2012_256, описанный в 4.1.1. Результатом работы данного преобразования является значение функции $PRF_{IPSEC\_KEYMAT\_256}$, аргументами которой являются байтовые строки K и S.

$$PRF_{IPSEC\_KEYMAT\_256}(K,S) = T_1 | T_2 | T_3 | \ldots, \tag{9}$$

где параметры T_i определяют последовательно следующим образом:

$$\begin{aligned} T_1 &= HMAC_{256}(K,\ S), \\ T_i &= HMAC_{256}(K,\ T_{i-1} | S), \end{aligned} \tag{10}$$

где индекс i изменяется в пределах, обеспечивающих выработку последовательности требуемой длины, определяемой протоколом.

Функция $PRF_{IPSEC\_KEYMAT\_256}(K,S)$ использует функцию $HMAC_{256}$, описанную в 4.1.1, и по схеме задания аргументов в итерациях аналогична функции $KEYMAT$ в [3].

4.2.2.2 PRF_IPSEC_PRFPLUS_GOSTR3411_2012_256

Преобразование PRF_IPSEC_PRFPLUS_GOSTR3411_2012_256 на основе хэш-функции H с длиной выхода, равной 256 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8), задает псевдослучайную функцию выработки ключевого материала протокола IPsec, использующую алгоритм HMAC_GOSTR3411_2012_256, описанный в 4.1.1. Результатом работы данного преобразования является значение функции $PRF_{IPSEC\_PRFPLUS\_256}$, аргументами которой являются байтовые строки K и S.

$$PRF_{IPSE\_PRFPLUS\_256}(K,S) = T_1 | T_2 | T_3 | \ldots, \tag{11}$$

где параметры T_i определяют последовательно следующим образом:

$$\begin{aligned} T_1 &= HMAC_{256}(K, S | 0x01), \\ T_i &= HMAC_{256}(K,\ T_{i-1} | S | i), \end{aligned} \tag{12}$$

где индекс i изменяется в пределах, обеспечивающих выработку последовательности требуемой длины, определяемой протоколом и не превышающей 255 × 256 бит, что соответствует выходной последовательности $T_1 | T_2 | T_3 | \ldots | T_{255}$.

Функция $PRF_{IPSEC\_PRFPLUS\_256}$ использует функцию $HMAC_{256}$, описанную в 4.1.1, и по схеме задания аргументов в итерациях аналогична функции *prf+* в [6].

4.2.3 Псевдослучайные функции протокола IPsec на основе ГОСТ Р 34.11, 512 бит

4.2.3.1 PRF_IPSEC_KEYMAT_GOSTR3411_2012_512

Преобразование PRF_IPSEC_KEYMAT_GOSTR3411_2012_512 на основе хэш-функции H с длиной выхода, равной 512 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8), задает псевдослучайную функцию выработки ключевого материала протокола IPsec, использующую алгоритм HMAC_GOSTR3411_2012_512, описанный в 4.1.2. Результатом работы данного преобразования является значение функции $PRF_{IPSEC\_KEYMAT\_512}$, аргументами которой являются байтовые строки K и S.

$$PRF_{IPSEC\_KEYMAT\_512}(K,S) = T_1 | T_2 | T_3 | \ldots, \quad (13)$$

где параметры T_i определяют последовательно следующим образом:

$$\begin{aligned} T_1 &= HMAC_{512}(K, S), \\ T_i &= HMAC_{512}(K, T_{i-1} | S), \end{aligned} \quad (14)$$

где индекс i изменяется в пределах, обеспечивающих выработку последовательности требуемой длины, определяемой протоколом.

Функция $PRF_{IPSEC\_KEYMAT\_512}(K,S)$ использует функцию $HMAC_{512}$, описанную в 4.1.2, и по схеме задания аргументов в итерациях аналогична функции $KEYMAT$ в [3].

4.2.3.2 PRF_IPSEC_PRFPLUS_GOSTR3411_2012_512

Преобразование PRF_IPSEC_PRFPLUS_GOSTR3411_2012_512 на основе хэш-функции H с длиной выхода, равной 512 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8), задает псевдослучайную функцию выработки ключевого материала протокола IPsec, использующую алгоритм HMAC_GOSTR3411_2012_512, описанный в 4.1.2. Результатом работы данного преобразования является значение функции $PRF_{IPSEC\_PRFPLUS\_512}$, аргументами которой являются байтовые строки K и S.

$$PRF_{IPSEC\_PRFPLUS\_512}(K,S) = T_1 | T_2 | T_3 | \ldots, \quad (11)$$

где параметры T_i определяют последовательно следующим образом:

$$\begin{aligned} T_1 &= HMAC_{512}(K, S | 0x01), \\ T_i &= HMAC_{512}(K, T_{i-1} | S | i), \end{aligned} \quad (12)$$

где индекс i изменяется в пределах, обеспечивающих выработку последовательности требуемой длины, определяемой протоколом и не превышающей 255 × 512 бит, что соответствует выходной последовательности $T_1 | T_2 | T_3 | \ldots | T_{255}$.

Функция $PRF_{IPSEC\_PRFPLUS\_512}$ использует функцию $HMAC_{512}$, описанную в 4.1.2, и по схеме задания аргументов в итерациях аналогична функции *prf+* в [6].

4.3 Алгоритмы согласования ключей VKO

В настоящем подразделе определены алгоритмы согласования с использованием ключей, определенных в соответствии с ГОСТ Р 34.10–2012 (раздел 5).

4.3.1 VKO_GOSTR3410_2012_256

VKO_GOSTR3410_2012_256 является алгоритмом согласования ключей VKO с длиной 256 бит на основе хэш-функции H с длиной выхода, равной 256 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8). Алгоритм можно выполнять с использованием ключей, определенных в соответствии с ГОСТ Р 34.10–2012 (раздел 5), длиной 256 и 512 бит.

Данный алгоритм предназначен для получения ключа шифрования либо ключевого материала длины 256 бит, далее используемых в криптографических протоколах. Ключ либо ключевой материал, обозначаемые $KEK_{VKO}(x, y, UKM)$, вырабатываются стороной обмена из своего закрытого ключа x, открытого ключа $y \cdot P$ противоположной стороны и величины UKM, рассматриваемой как число.

Алгоритм можно использовать как для статических, так и для эфемерных ключей сторон при битовой длине открытого ключа n, где $n \geq 512$, в том числе и для случая, когда ключи одной из сторон являются статическими, а другой – эфемерными.

UKM используют опционально (иначе величина UKM полагается равной 1) и принимает значение от 1 до $2^{n/2} - 1$. Допускается осуществлять выбор ненулевого значения UKM любой битовой длины, не превосходящей $n/2$. Использование UKM с битовой длиной не менее 64 рекомендуется в том случае, когда ключи хотя бы одной из сторон являются статическими.

$$K\ (x, y, UKM)\ =\ (m/q \cdot UKM \cdot x\ mod\ q) \cdot (y \cdot P), \tag{17}$$

где m и q – параметры используемой эллиптической кривой, соответствующие обозначениям, принятым в разделе 5 ГОСТ Р 34.10–2012 (m – порядок группы точек эллиптической кривой, q – порядок циклической подгруппы), P – ненулевая точка подгруппы; P задается протоколом.

$$KEK_{VKO}\ (x, y, UKM)\ =\ H_{256}\ (K\ (x, y, UKM)). \tag{18}$$

Данный алгоритм определяют по аналогии с подразделом 5.2 в [4], используя вместо хэш-функции h, определенной в ГОСТ Р 34.11–94 (раздел 6, обозначена в [4] как *gostR*3411), хэш-функцию H_{256} и вычисляя $K\ (x, y, UKM)$ при длинах открытых ключей $n \geq\ 512$ бит и длине UKM – до $n/2$ бит.

4.3.2 VKO_GOSTR3410_2012_512

VKO_GOSTR3410_2012_512 является алгоритмом согласования ключей VKO с длиной 512 бит на основе хэш-функции H с длиной выхода, равной 512 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8). Алгоритм можно выполнять с использованием ключей, определенных в соответствии с ГОСТ Р 34.10–2012 (раздел 5), длиной 512 бит.

Данный алгоритм предназначен для получения ключа шифрования либо ключевого материала длины 512 бит, далее используемых в криптографических протоколах. Ключ либо ключевой материал, обозначаемые $KEK_{VKO}\ (x, y, UKM)$,

вырабатываются стороной обмена из своего закрытого ключа x, открытого ключа $y \cdot P$ противоположной стороны и величины UKM, рассматриваемой как число.

Алгоритм можно использовать как для статических, так и для эфемерных ключей сторон при битовой длине открытого ключа n, где $n \geq 1024$, в том числе и для случая, когда ключи одной из сторон являются статическими, а другой – эфемерными.

UKM используется опционально (иначе величина UKM полагается равной 1) и принимает значение от 1 до $2^{n/2} - 1$. Допускается осуществлять выбор ненулевого значения UKM любой битовой длины, не превосходящей $n/2$. Использование UKM с битовой длиной не менее 128 рекомендуется в случае, когда ключи хотя бы одной из сторон являются статическими.

$$K\,(x, y, UKM) = (m/q \cdot UKM \cdot x \bmod q) \cdot (y \cdot P)\,, \tag{19}$$

где m и q – параметры используемой эллиптической кривой, соответствующие обозначениям, принятым в разделе 5 ГОСТ Р 34.10–2012 (m – порядок группы точек эллиптической кривой, q – порядок циклической подгруппы), P – ненулевая точка подгруппы; P задается протоколом.

$$KEK_{VKO}\,(x, y, UKM) = H_{512}\,(K\,(x, y, UKM)). \tag{20}$$

Данный алгоритм определяют по аналогии с подразделом 5.2 в [4], используя вместо хэш-функции h, определенной в ГОСТ Р 34.11–94 (раздел 6, обозначена в [4] как *gostR*3411), хэш-функцию H_{512} и вычисляя $K\,(x, y, UKM)$ при длинах открытых ключей $n \geq 1024$ бит и длине UKM – до $n/2$ бит.

4.4 Алгоритм диверсификации KDF_GOSTR3411_2012_256

Алгоритм KDF_GOSTR3411_2012_256 на основе хэш-функции H с длиной выхода, равной 256 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8), задает функцию диверсификации для порождения ключевого материала длиной 256 бит, использующую алгоритм HMAC_GOSTR3411_2012_256, описанный в 4.1.1. Результатом работы данного алгоритма является значение функции KDF_{256}, аргументами которой являются байтовые строки K_{in}, $label$ и $seed$.

$$KDF_{256}\,(K_{in}, label, seed) = HMAC_{256}\,(K_{in}, 0x01 \mid label \mid 0x00 \mid seed \mid 0x01 \mid 0x00), \tag{21}$$

где

K_{in} – ключ диверсификации;

$label, seed$ – параметры, задаваемые протоколом, их длины должны быть фиксированы протоколом.

Алгоритм диверсификации KDF_GOSTR3411_2012_256 является частным случаем алгоритма KDF_TREE_GOSTR3411_2012_256, описанной в следующем подразделе.

4.5 Алгоритм диверсификации KDF_TREE_GOSTR3411_2012_256

Алгоритм KDF_TREE_GOSTR3411_2012_256 на основе хэш-функции H с длиной выхода, равной 256 битам, определенной в ГОСТ Р 34.11–2012 (раздел 8), задает функцию диверсификации, использующую алгоритм HMAC_GOSTR3411_2012_256, описанный в 4.1.1. Результатом работы данного алгоритма является значение функции KDF_{TREE_256}, аргументами которой являются байтовые строки K_{in}, $label$, $seed$ и R.

$$KDF_{TREE\_256}\,(K_{in}, label, seed, R) = \text{К}(1) \mid \text{К}(2) \mid \text{К}(3) \mid \text{К}(4) \mid ...; \qquad (22)$$

$$K\,(i) = HMAC_{256}\,(K_{in}, [i]_b \mid label \mid 0x00 \mid seed \mid [L]_b)\,, \qquad i \geq 1 \qquad (23)$$

где

K_{in} – ключ диверсификации;

$label, seed$ – параметры, задаваемые протоколом, их длины должны быть фиксированы протоколом;

R – внешний фиксируемый параметр, с возможными значениями 1, 2, 3, 4;

i – счетчик числа итераций;

$[i]_b$ – байтовое представление счетчика числа итераций, количество байт в представлении $[i]_b$ равно значению R (не более 4 байт), записывается в сетевом порядке байт;

L – необходимая битовая длина вырабатываемого ключевого материала (целое число), не превосходящее $256 \cdot (2^{8\text{R}} - 1)$;

$[L]_b$ – байтовое представление L, записывается в сетевом порядке минимально необходимого числа байт.

Алгоритм диверсификации KDF_TREE_GOSTR3411_2012_256 предназначен для порождения ключевого материала длины L, не превосходящей $256 \cdot (2^{8\text{R}} - 1)$ бит, и использует общие принципы задания входных параметров и выхода для функций диверсификации, изложенные в 5.1 [7]. В качестве псевдослучайной функции выбрана функция $HMAC_{256}$, описанная в 4.1.1.

Если $R = 1$ и $L = 256$, то алгоритм KDF_TREE_GOSTR3411_2012_256 совпадает с алгоритмом KDF_GOSTR3411_2012_256 из предыдущего подраздела.

Каждый ключ, последовательно полученный из ключевого материала, сформированного с помощью ключа диверсификации K_{in} – ключа 0-го уровня, можно затем рассматривать как ключ диверсификации 1-го уровня и также использовать для генерации ключевого материала. Ключевой материал, полученный из ключа диверсификации 1-го уровня, может быть разбит на ключи диверсификации 2-го уровня. Применение данной процедуры приводит к построению ключевого дерева с корневым ключом K_{in} и формированию ключевого материала с иерархией по уровням, как описано

в разделе 6 [7]. Процедура разбиения ключевого материала на каждом уровне определена в протоколах.

4.6 Экспорт и импорт ключей

При экспорте секретного ключа K с использованием заданного ключа экспорта K_e и случайного набора $seed$ длины от 8 до 16 байт формируется экспортное представление ключа K по следующей схеме:

1) Порождается случайный набор $seed$.

2) С помощью функции диверсификации, использующей в качестве ключа диверсификации ключ экспорта K_e, производится формирование ключа $KEK_e(seed)$:

$$KEK_e\,(seed) = KDF_{256}(K_e, label, seed), \tag{24}$$

где в качестве функции диверсификации использована функция KDF_{256}, описанная в 4.4, при фиксированном значении

$$label = (0x26 \mid 0xBD \mid 0xB8 \mid 0x78).$$

3) Вычисляется значение имитовставки в соответствии с разделом 5 ГОСТ 28147–89 длины 4 байта от данных K на ключе $KEK_e\,(seed)$, синхропосылка при этом полагается равной первым 8 байтам $seed$. Полученный набор обозначается через CEK_MAC.

4) Ключ K зашифровывается в соответствии с разделом 2 ГОСТ 28147–89 в режиме простой замены с использованием ключа $KEK_e\,(seed)$. Результат зашифрования обозначается через CEK_ENC.

5) Экспортным представлением ключа полагается набор $(seed \mid CEK\_ENC \mid CEK\_MAC)$.

При импорте ключа по экспортному представлению ключа и ключу экспорта K_e восстанавливается ключ K по следующей схеме:

1) Из экспортного представления ключа выделяются наборы $seed$, CEK_ENC и CEK_MAC.

2) С помощью функции диверсификации, использующей в качестве ключа диверсификации ключ экспорта K_e, производится формирование ключа, обозначаемого $KEK_e(seed)$:

$$KEK_e\,(seed) = KDF_{256}(K_e, label, seed), \tag{25}$$

где в качестве функции диверсификации использована функция KDF_{256}, описанная в 4.4, при фиксированном значении

$$label = (0x26 \mid 0xBD \mid 0xB8 \mid 0x78).$$

3) Набор CEK_ENC расшифровывается в соответствии с разделом 2 по алгоритму ГОСТ 28147–89 в режиме простой замены с использованием ключа $KEK_e\,(seed)$. Ключ K полагается равным результату расшифрования.

4) Вычисляется значение имитовставки в соответствии с разделом 5 ГОСТ 28147–89 длины 4 байта от данных K на ключе $KEK_e\,(seed)$, синхропосылка при этом полагается равной первым 8 байтам $seed$. Если результат отличен от CEK_MAC, возвращается ошибка.

Данные алгоритмы экспорта и импорта ключей являются модификациями алгоритмов CryptoPro Key Wrap и CryptoPro Key Unwrap, описанных в 6.3 и 6.4 [4].

Приложение А
(справочное)

Контрольные примеры

Данное приложение носит справочный характер и не является частью настоящих рекомендаций.

1) HMAC_GOSTR3411_2012_256
Ключ K:

00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f

Данные T:

01 26 bd b8 78 00 af 21 43 41 45 65 63 78 01 00

Значение $HMAC_{256}\ (K,T)$:

a1 aa 5f 7d e4 02 d7 b3 d3 23 f2 99 1c 8d 45 34
01 31 37 01 0a 83 75 4f d0 af 6d 7c d4 92 2e d9

2) HMAC_GOSTR3411_2012_512
Ключ K:

00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f

Данные T:

01 26 bd b8 78 00 af 21 43 41 45 65 63 78 01 00

Значение $HMAC_{512}\ (K,T)$:

a5 9b ab 22 ec ae 19 c6 5f bd e6 e5 f4 e9 f5 d8
54 9d 31 f0 37 f9 df 9b 90 55 00 e1 71 92 3a 77
3d 5f 15 30 f2 ed 7e 96 4c b2 ee dc 29 e9 ad 2f
3a fe 93 b2 81 4f 79 f5 00 0f fc 03 66 c2 51 e6

3) PRF_TLS_GOSTR3411_2012_256
Ключ K:

00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f

10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f

Данные $seed$:

18 47 1d 62 2d c6 55 c4 d2 d2 26 96 91 ca 4a 56
0b 50 ab a6 63 55 3a f2 41 f1 ad a8 82 c9 f2 9a

Данные $label$:

11 22 33 44 55

Выход T_1:

ff 09 66 4a 44 74 58 65 94 4f 83 9e bb 48 96 5f
15 44 ff 1c c8 e8 f1 6f 24 7e e5 f8 a9 eb e9 7f

Выход T_2:

c4 e3 c7 90 0e 46 ca d3 db 6a 01 64 30 63 04 0e
c6 7f c0 fd 5c d9 f9 04 65 23 52 37 bd ff 2c 02

4) PRF_TLS_GOSTR3411_2012_512
Ключ K:

00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f

Данные $seed$:

18 47 1d 62 2d c6 55 c4 d2 d2 26 96 91 ca 4a 56
0b 50 ab a6 63 55 3a f2 41 f1 ad a8 82 c9 f2 9a

Данные $label$:

11 22 33 44 55

Выход T_1:

f3 51 87 a3 dc 96 55 11 3a 0e 84 d0 6f d7 52 6c
5f c1 fb de c1 a0 e4 67 3d d6 d7 9d 0b 92 0e 65
ad 1b c4 7b b0 83 b3 85 1c b7 cd 8e 7e 6a 91 1a
62 6c f0 2b 29 e9 e4 a5 8e d7 66 a4 49 a7 29 6d

Выход T_2:

e6 1a 7a 26 c4 d1 ca ee cf d8 0c ca 65 c7 1f 0f
88 c1 f8 22 c0 e8 c0 ad 94 9d 03 fe e1 39 57 9f
72 ba 0c 3d 32 c5 f9 54 f1 cc cd 54 08 1f c7 44
02 78 cb a1 fe 7b 7a 17 a9 86 fd ff 5b d1 5d 1f

5) PRF_IPSEC_KEYMAT_GOSTR3411_2012_256
Ключ K:

c9 a9 a7 73 20 e2 cc 55 9e d7 2d ce 6f 47 e2 19
2c ce a9 5f a6 48 67 05 82 c0 54 c0 ef 36 c2 21

Данные S:

01 26 bd b8 78 00 1d 80 60 3c 85 44 c7 27 01 00

Выход T_1:

21 01 d8 0c 47 db 54 bc 3c 82 9b 8c 30 7c 47 55
50 88 83 a6 d6 9e 60 1b f7 aa fb 0a bc a4 ed 95

Выход T_2:

33 b8 4e d0 8f 93 56 f8 1d f8 d2 79 f0 79 c9 02
87 cb 45 2c 81 d4 1e 80 38 43 08 86 c1 92 12 aa

6) PRF_IPSEC_PRFPLUS_GOSTR3411_2012_256
Ключ K:

c9 a9 a7 73 20 e2 cc 55 9e d7 2d ce 6f 47 e2 19
2c ce a9 5f a6 48 67 05 82 c0 54 c0 ef 36 c2 21

Данные S:

01 26 bd b8 78 00 1d 80 60 3c 85 44 c7 27 01 00

Выход T_1:

2d e5 ee 84 e1 3d 7b e5 36 16 67 39 13 37 0a b0
54 c0 74 b7 9b 69 a8 a8 46 82 a9 f0 4f ec d5 87

Выход T_2:

29 f6 0d da 45 7b f2 19 aa 2e f9 5d 7a 59 be 95
4d e0 08 f4 a5 0d 50 4d bd b6 90 be 68 06 01 53

7) PRF_IPSEC_KEYMAT_GOSTR3411_2012_512
Ключ K:

c9 a9 a7 73 20 e2 cc 55 9e d7 2d ce 6f 47 e2 19
2c ce a9 5f a6 48 67 05 82 c0 54 c0 ef 36 c2 21

Данные S:

01 26 bd b8 78 00 1d 80 60 3c 85 44 c7 27 01 00

Выход T_1:

b9 55 5b 29 91 75 4b 37 9d a6 8e 60 98 f5 b6 0e
df 91 8a 56 20 4b ff f3 a8 37 6d 1f 57 ed b2 34
a5 12 32 81 23 cd 6c 03 0b 54 14 2e 1e c7 78 2b
03 00 be a5 7c c2 a1 4c a3 b4 f0 85 a4 5c d6 ca

Выход T_2:

37 b1 e0 86 52 43 a4 fb 29 14 8d 27 4d 30 63 fc
bf b0 f2 f4 68 d5 27 e4 3b ca 41 fa 6b b5 3e c8
df 21 bf c4 62 3a 2e 76 8b 64 54 03 3e 09 52 32
d1 8c 86 a6 8f 00 98 d3 31 81 75 f6 59 05 ae db

8) PRF_IPSEC_ PRFPLUS_GOSTR3411_2012_512
Ключ K:

c9 a9 a7 73 20 e2 cc 55 9e d7 2d ce 6f 47 e2 19
2c ce a9 5f a6 48 67 05 82 c0 54 c0 ef 36 c2 21

Данные S:

01 26 bd b8 78 00 1d 80 60 3c 85 44 c7 27 01 00

Выход T_1:

5d a6 71 43 a5 f1 2a 6d 6e 47 42 59 6f 39 24 3f
cc 61 57 45 91 5b 32 59 10 06 ff 78 a2 08 63 d5
f8 8e 4a fc 17 fb be 70 b9 50 95 73 db 00 5e 96
26 36 98 46 cb 86 19 99 71 6c 16 5d d0 6a 15 85

Выход T_2:

48 34 49 5a 43 74 6c b5 3f 0a ba 3b c4 6e bc f8
77 3c a6 4a d3 43 c1 22 ee 2a 57 75 57 03 81 57
ee 9c 38 8d 96 ef 71 d5 8b e5 c1 ef a1 af a9 5e
be 83 e3 9d 00 e1 9a 5d 03 dc d6 0a 01 bc a8 e3

9) VKO_GOSTR3410_2012_256 с выходом 256 на ключах ГОСТ Р 34.10–2012, 512 бит, на параметрах id-tc26-gost-3410-12-512-paramSetA
Величина UKM:

1d 80 60 3c 85 44 c7 27

Закрытый ключ x стороны *A*:

c9 90 ec d9 72 fc e8 4e c4 db 02 27 78 f5 0f ca
c7 26 f4 67 08 38 4b 8d 45 83 04 96 2d 71 47 f8
c2 db 41 ce f2 2c 90 b1 02 f2 96 84 04 f9 b9 be
6d 47 c7 96 92 d8 18 26 b3 2b 8d ac a4 3c b6 67

Открытый ключ $x \cdot P$ стороны *A* (точка кривой (*X*,*Y*)):

aa b0 ed a4 ab ff 21 20 8d 18 79 9f b9 a8 55 66
54 ba 78 30 70 eb a1 0c b9 ab b2 53 ec 56 dc f5
d3 cc ba 61 92 e4 64 e6 e5 bc b6 de a1 37 79 2f
24 31 f6 c8 97 eb 1b 3c 0c c1 43 27 b1 ad c0 a7
91 46 13 a3 07 4e 36 3a ed b2 04 d3 8d 35 63 97
1b d8 75 8e 87 8c 9d b1 14 03 72 1b 48 00 2d 38
46 1f 92 47 2d 40 ea 92 f9 95 8c 0f fa 4c 93 75
64 01 b9 7f 89 fd be 0b 5e 46 e4 a4 63 1c db 5a

Закрытый ключ y стороны *B*:

48 c8 59 f7 b6 f1 15 85 88 7c c0 5e c6 ef 13 90
cf ea 73 9b 1a 18 c0 d4 66 22 93 ef 63 b7 9e 3b
80 14 07 0b 44 91 85 90 b4 b9 96 ac fe a4 ed fb
bb cc cc 8c 06 ed d8 bf 5b da 92 a5 13 92 d0 db

Открытый ключ $y \cdot P$ стороны *B* (точка кривой (*X*,*Y*)):

19 2f e1 83 b9 71 3a 07 72 53 c7 2c 87 35 de 2e
a4 2a 3d bc 66 ea 31 78 38 b6 5f a3 25 23 cd 5e
fc a9 74 ed a7 c8 63 f4 95 4d 11 47 f1 f2 b2 5c
39 5f ce 1c 12 91 75 e8 76 d1 32 e9 4e d5 a6 51
04 88 3b 41 4c 9b 59 2e c4 dc 84 82 6f 07 d0 b6
d9 00 6d da 17 6c e4 8c 39 1e 3f 97 d1 02 e0 3b

b5 98 bf 13 2a 22 8a 45 f7 20 1a ba 08 fc 52 4a
2d 77 e4 3a 36 2a b0 22 ad 40 28 f7 5b de 3b 79

Значение KEK_{VKO}:

c9 a9 a7 73 20 e2 cc 55 9e d7 2d ce 6f 47 e2 19
2c ce a9 5f a6 48 67 05 82 c0 54 c0 ef 36 c2 21

10) VKO_GOSTR3410_2012_512 с выходом 512 на ключах ГОСТ Р 34.10–2012, 512 бит, на параметрах id-tc26-gost-3410-12-512-paramSetA
Величина $UKM:$

1d 80 60 3c 85 44 c7 27

Закрытый ключ x стороны *A*:

c9 90 ec d9 72 fc e8 4e c4 db 02 27 78 f5 0f ca
c7 26 f4 67 08 38 4b 8d 45 83 04 96 2d 71 47 f8
c2 db 41 ce f2 2c 90 b1 02 f2 96 84 04 f9 b9 be
6d 47 c7 96 92 d8 18 26 b3 2b 8d ac a4 3c b6 67

Открытый ключ $x \cdot P$ стороны *A* (точка кривой (*X*, *Y*)):

aa b0 ed a4 ab ff 21 20 8d 18 79 9f b9 a8 55 66
54 ba 78 30 70 eb a1 0c b9 ab b2 53 ec 56 dc f5
d3 cc ba 61 92 e4 64 e6 e5 bc b6 de a1 37 79 2f
24 31 f6 c8 97 eb 1b 3c 0c c1 43 27 b1 ad c0 a7
91 46 13 a3 07 4e 36 3a ed b2 04 d3 8d 35 63 97
1b d8 75 8e 87 8c 9d b1 14 03 72 1b 48 00 2d 38
46 1f 92 47 2d 40 ea 92 f9 95 8c 0f fa 4c 93 75
64 01 b9 7f 89 fd be 0b 5e 46 e4 a4 63 1c db 5a

Закрытый ключ y стороны *B*:

48 c8 59 f7 b6 f1 15 85 88 7c c0 5e c6 ef 13 90
cf ea 73 9b 1a 18 c0 d4 66 22 93 ef 63 b7 9e 3b
80 14 07 0b 44 91 85 90 b4 b9 96 ac fe a4 ed fb
bb cc cc 8c 06 ed d8 bf 5b da 92 a5 13 92 d0 db

Открытый ключ $y \cdot P$ стороны *B* (точка кривой (*X*, *Y*)):

19 2f e1 83 b9 71 3a 07 72 53 c7 2c 87 35 de 2e
a4 2a 3d bc 66 ea 31 78 38 b6 5f a3 25 23 cd 5e
fc a9 74 ed a7 c8 63 f4 95 4d 11 47 f1 f2 b2 5c

39 5f ce 1c 12 91 75 e8 76 d1 32 e9 4e d5 a6 51
04 88 3b 41 4c 9b 59 2e c4 dc 84 82 6f 07 d0 b6
d9 00 6d da 17 6c e4 8c 39 1e 3f 97 d1 02 e0 3b
b5 98 bf 13 2a 22 8a 45 f7 20 1a ba 08 fc 52 4a
2d 77 e4 3a 36 2a b0 22 ad 40 28 f7 5b de 3b 79

Значение KEK_{VKO}:

79 f0 02 a9 69 40 ce 7b de 32 59 a5 2e 01 52 97
ad aa d8 45 97 a0 d2 05 b5 0e 3e 17 19 f9 7b fa
7e e1 d2 66 1f a9 97 9a 5a a2 35 b5 58 a7 e6 d9
f8 8f 98 2d d6 3f c3 5a 8e c0 dd 5e 24 2d 3b df

11) Алгоритм диверсификации KDF_GOSTR3411_2012_256:
Ключ K_{in}:

00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f

$label:$

26 bd b8 78

$seed:$

af 21 43 41 45 65 63 78

Значение $KDF_{256}\ (K_{in}, label, seed)$:

a1 aa 5f 7d e4 02 d7 b3 d3 23 f2 99 1c 8d 45 34
01 31 37 01 0a 83 75 4f d0 af 6d 7c d4 92 2e d9

12) Алгоритм диверсификации KDF_TREE_GOSTR3411_2012_256
Длина выхода L:

512

Ключ K_{in}:

00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f

$label$:

26 bd b8 78

$seed:$

af 21 43 41 45 65 63 78

Значение K_1:

22 b6 83 78 45 c6 be f6 5e a7 16 72 b2 65 83 10
86 d3 c7 6a eb e6 da e9 1c ad 51 d8 3f 79 d1 6b

Значение K_2:

07 4c 93 30 59 9d 7f 8d 71 2f ca 54 39 2f 4d dd
e9 37 51 20 6b 35 84 c8 f4 3f 9e 6d c5 15 31 f9

Значение параметра $R:$

1

13) Экспорт и импорт ключей на параметрах szOID_Gost28147_89_TC26_Z_ParamSet
Ключ K_e:

00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f

Ключ K:

20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f
30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f

Величина $seed$:

af 21 43 41 45 65 63 78

$label:$

26 bd b8 78

$KEK_e(seed) = KDF_{256}(K_e, label, seed):$

a1 aa 5f 7d e4 02 d7 b3 d3 23 f2 99 1c 8d 45 34
01 31 37 01 0a 83 75 4f d0 af 6d 7c d4 92 2e d9

CEK_MAC:

be 33 f0 52

CEK_ENC:

d1 55 47 f8 ee 85 12 1b c8 7d 4b 10 27 d2 60 27
ec c0 71 bb a6 e7 2f 3f ec 6f 62 0f 56 83 4c 5a

Библиография

[1] RFC2104 — Х. Кравчик, М. Белларе и Р. Канетти «HMAC: ключевое хэширование для проверки подлинности сообщений» (H. Krawczyk, M. Bellare and R. Canetti, HMAC: Keyed-Hashing for Message Authentication, Informational, IETF RFC 2104, February 1997)

[2] RFC2246 — К. Аллен, Т. Диркс «Протокол TLS – Transport Layer Security – версия 1.0» (C. Allen, T. Dierks, The TLS Protocol Version 1.0, Standards Track, IETF RFC 2246, January 1999)

[3] RFC2409 — Д. Харкинс, Д. Каррел «Протокол обмена ключами в сети Интернет (IKE)» (Harkins, D. and D. Carrel, The Internet Key Exchange (IKE), Standards Track, IETF RFC 2409, November 1998)

[4] RFC4357 — В. Попов, И. Курепкин, С. Леонтьев «Дополнительные алгоритмы шифрования для использования с алгоритмами по ГОСТ 28147–89, ГОСТ Р 34.10–94, ГОСТ Р 34.10–2001 и ГОСТ Р 34.11–94» (Popov V., Kurepkin I. and S. Leontiev, Additional Cryptographic Algorithms for Use with GOST 28147–89, GOST R 34.10–94, GOST R 34.10–2001, and GOST R 34.11–94 Algorithms, Informational, IETF RFC 4357, January 2006)

[5] RFC5246 — Т. Диркс, Е. Рескорла «Протокол TLS – Transport Layer Security – версия 1.0» (T. Dierks, E. Rescorla, The Transport Layer Security (TLS) Protocol Version 1.2, Standards Track, IETF RFC 5246, August 2008)

[6] RFC7296 — С. Кауфман, П. Хофман, Й. Нир, П. Еронен «Протокол обмена ключами в сети Интернет, версия 2 (IKEv2)» (Kaufman C., Hoffman P., Nir Y., and P. Eronen, Internet Key Exchange Protocol Version 2 (IKEv2), Standards Track, IETF RFC 7296, October 2014)

[7] NIST SP 800-108 — Лили Чен «Рекомендации по получению производных ключей с использованием псевдослучайных функций», Национальный институт стандартов и технологий США (Lily Chen, Recommendation for Key Derivation Using Pseudorandom Functions, National Institute of Standards and Technology (NIST) Special Publication 800-108, October 2009)

УДК 681.3.06:006.354 ОКС 35. 040 ОКСТУ 5002 П85

Ключевые слова: криптографическая защита информации, криптографические алгоритмы, ключ