ГОСУДАРСТВЕННЫЙ СТАНДАРТ
РЕСПУБЛИКИ БЕЛАРУСЬ

СТБ 34.101.47-2012

Информационные технологии и безопасность

КРИПТОГРАФИЧЕСКИЕ АЛГОРИТМЫ ГЕНЕРАЦИИ ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ

Інфармацыйныя тэналогіі і бяспека

КРЫПТАГРАФІЧНЫЯ АЛГАРЫТМЫ ГЕНЕРАЦЫІ ПСЕЎДАВЫПАДКОВЫХ ЛІКАЎ

Госстандарт
Минск

УДК МКС 35.240.40 КП 05

Ключевые слова: псевдослучайные числа, ключ, синхропосылка, хэширование

Предисловие

Цели, основные принципы, положения по государственному регулированию и управлению в области технического нормирования и стандартизации установлены Законом Республики Беларусь «О техническом нормировании и стандартизации».

1 РАЗРАБОТАН закрытым акционерным обществом «АВЕСТ»

ВНЕСЕН Национальным банком Республики Беларусь

2 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ постановлением Госстандарта Республики Беларусь от 20 апреля 2012 г. № 21

3 ВВЕДЕН ВПЕРВЫЕ

Издан на русском языке

Содержание

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РЕСПУБЛИКИ БЕЛАРУСЬ

Информационные технологии и безопасность

КРИПТОГРАФИЧЕСКИЕ АЛГОРИТМЫ ГЕНЕРАЦИИ ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ

Інфармацыйныя тэхналогіі і бяспека

КРЫПТАГРАФІЧНЫЯ АЛГАРЫТМЫ ГЕНЕРАЦЫІ ПСЕЎДАВЫПАДКОВЫХ ЛІКАЎ

Information technology and security

Cryptographic algorithms of pseudorandom number generation

Дата введения 2012-08-01

1 Область применения

Настоящий стандарт устанавливает криптографические алгоритмы генерации псевдослучайных чисел. Стандарт предназначен для получения ключей, синхропосылок, других непредсказуемых и уникальных параметров криптографических алгоритмов и протоколов.

Настоящий стандарт применяется при разработке, испытаниях и эксплуатации средств криптографической защиты информации.

2 Нормативные ссылки

В настоящем стандарте использованы ссылки на следующие технические нормативные правовые акты в области технического нормирования и стандартизации (далее — ТНПА):

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

СТБ 34.101.19-2011 Информационные технологии. Форматы сертификатов и списков отозванных сертификатов инфраструктуры открытых ключей

СТБ 34.101.31-2011 Информационные технологии и безопасность. Криптографические алгоритмы шифрования и контроля целостности

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

Примечание — При пользовании настоящим стандартом целесообразно проверить действие ТНПА по каталогу, составленному по состоянию на 1 января текущего года, и по соответствующим информационным указателям, опубликованным в текущем году.

Если ссылочные ТНПА заменены (изменены), то при пользовании настоящим стандартом следует руководствоваться замененными (измененными) ТНПА. Если ссылочные ТНПА отменены без замены, то положение, в котором дана ссылка на них, применяется в части, не затрагивающей эту ссылку.

3 Термины и определения

В настоящем стандарте применяются следующие термины с соответствующими определениями:

3.1 имитовставка: Двоичное слово, которое определяется по сообщению с использованием ключа и служит для контроля целостности и подлинности сообщения.

3.2 ключ: Параметр, который управляет криптографическими операциями зашифрования и расшифрования, выработки и проверки электронной цифровой подписи, генерации псевдослучайных чисел и др.

3.3 октет: Двоичное слово длины 8.

3.4 псевдослучайные числа: Последовательность элементов, полученная в результате выполнения некоторого алгоритма и используемая в конкретном случае вместо последовательности случайных чисел.

3.5 синхропосылка: Открытые входные данные криптографического алгоритма, которые обеспечивают уникальность результатов криптографического преобразования на фиксированном ключе.

3.6 случайные числа: Последовательность элементов, каждый из которых не может быть предсказан (вычислен) только на основе знания предшествующих ему элементов данной последовательности.

3.7 сообщение: Двоичное слово конечной длины.

3.8 хэш-значение: Двоичное слово фиксированной длины, которое определяется по сообщению без использования ключа и служит для контроля целостности сообщения и для представления сообщения в сжатой форме.

3.9 хэширование: Выработка хэш-значений.

4 Обозначения

4.1 Список обозначений

$\{0,1\}^n$	множество всех слов длины n в алфавите $\{0,1\}$;
$\{0,1\}^*$	множество всех слов конечной длины в алфавите $\{0,1\}$ (включая пустое слово длины 0);
$\{0,1\}^{n*}$	множество всех слов из $\{0,1\}^*$, длина которых кратна n;
$\lvert u\rvert$	длина слова $u \in \{0,1\}^*$;
α^n	слово длины n из одинаковых символов $\alpha \in \{0,1\}$;
$u \parallel v$	конкатенация $u_1u_2\dots u_nv_1v_2\dots v_m$ слов $u = u_1u_2\dots u_n$ и $v = v_1v_2\dots v_m$;
$\mathtt{01234...}_{16}$	представление $u \in \{0,1\}^{4*}$ шестнадцатеричным словом, при котором последовательным четырем символам u соответствует один шестнадцатеричный символ (например, $10100010 = \mathtt{A2}_{16}$);
$x \bmod m$	для целого x и натурального m остаток от деления x на m, т. е. число $r \in \{0,1,\dots,m-1\}$ такое, что m делит $x - r$;

$u \oplus v$	для $u = u_1u_2\dots u_n \in \{0,1\}^n$ и $v = v_1v_2\dots v_n \in \{0,1\}^n$ слово $w = w_1w_2\dots w_n \in \{0,1\}^n$ из символов $w_i = (u_i + v_i) \bmod 2$;
$\bar{u}$	а) для $u = u_1u_2\dots u_8 \in \{0,1\}^8$ число $2^7u_1 + 2^6u_2 + \dots + u_8$ и б) для $u = u_1 \parallel u_2 \parallel \dots \parallel u_n$, $u_i \in \{0,1\}^8$, число $\bar{u}_1 + 2^8\bar{u}_2 + \dots + 2^{8(n-1)}\bar{u}_n$;
$\langle U \rangle_{8n}$	для целого U слово $u \in \{0,1\}^{8n}$ такое, что $\bar{u} = U \bmod 2^{8n}$;
$u \boxplus v$	для $u, v \in \{0,1\}^{8n}$ слово $\langle \bar{u} + \bar{v} \rangle_{8n}$;
$c \leftarrow u$	присвоение переменной c значения u.

4.2 Пояснения к обозначениям

4.2.1 Слова

Двоичные слова представляют собой последовательности символов из алфавита $\{0,1\}$. Символы нумеруются слева направо от единицы. В настоящем подразделе в качестве примера рассматривается слово

$$w = 10110001100101001011101011001000.$$

В этом слове первый символ — 1, второй — 0, ..., последний — 0.

Слова разбиваются на тетрады из четверок последовательных двоичных символов. Тетрады кодируются шестнадцатеричными символами по следующим правилам (см. таблицу 1):

Таблица 1

тетрада	символ	тетрада	символ	тетрада	символ	тетрада	символ
0000	$\mathtt{0}_{16}$	0001	$\mathtt{1}_{16}$	0010	$\mathtt{2}_{16}$	0011	$\mathtt{3}_{16}$
0100	$\mathtt{4}_{16}$	0101	$\mathtt{5}_{16}$	0110	$\mathtt{6}_{16}$	0111	$\mathtt{7}_{16}$
1000	$\mathtt{8}_{16}$	1001	$\mathtt{9}_{16}$	1010	$\mathtt{A}_{16}$	1011	$\mathtt{B}_{16}$
1100	$\mathtt{C}_{16}$	1101	$\mathtt{D}_{16}$	1110	$\mathtt{E}_{16}$	1111	$\mathtt{F}_{16}$

Пары последовательных тетрад образуют октеты. Последовательные октеты слова w имеют вид:

$$10110001 = \mathtt{B1}_{16},\ 10010100 = \mathtt{94}_{16},\ 10111010 = \mathtt{BA}_{16},\ 11001000 = \mathtt{C8}_{16}.$$

4.2.2 Слова как числа

Октету $u = u_1u_2\dots u_8$ ставится в соответствие байт — число $\bar{u} = 2^7u_1 + 2^6u_2 + \dots + u_8$. Например, октетам w соответствуют байты

$$177 = 2^7 + 2^5 + 2^4 + 1,\ 148 = 2^7 + 2^4 + 2^2,\ 186 = 2^7 + 2^5 + 2^4 + 2^3 + 2^1,\ 200 = 2^7 + 2^6 + 2^3.$$

Число ставится в соответствие не только октетам, но и любому другому двоичному слову, длина которого кратна 8. При этом используется распространенное для многих современных процессоров соглашение «от младших к старшим» (little-endian): считается,

что первый байт является младшим, последний — старшим. Например, слову w соответствует число

$$\bar{w} = 177 + 2^8 \cdot 148 + 2^{16} \cdot 186 + 2^{24} \cdot 200 = 3367670961.$$

5 Общие положения

5.1 Назначение

Настоящий стандарт определяет криптографические алгоритмы генерации псевдослучайных чисел. В алгоритмах используются ключ и синхропосылка. При соблюдении секретности ключа и уникальности синхропосылки генерируемые числа нельзя предсказать или повторить, и поэтому их можно использовать для построения непредсказуемых и уникальных параметров криптографических алгоритмов и протоколов, в том числе других ключей и синхропосылок.

Ключ алгоритма генерации может быть известен одной или нескольким сторонам. В первом случае псевдослучайные числа можно использовать для построения секретных параметров владельца ключа. Во втором случае стороны могут использовать алгоритм для построения общих секретных параметров.

В 6.1 определяется алгоритм выработки имитовставки HMAC, соответствующий [1]. HMAC используется как вспомогательный алгоритм при генерации псевдослучайных чисел в 6.3. Кроме этого, HMAC имеет самостоятельное значение и может применяться непосредственно для выработки имитовставок. Если на вход HMAC подавать неповторящиеся сообщения, например отметки текущего времени, то выходные имитовставки можно использовать в качестве псевдослучайных чисел.

В 6.2 определяется алгоритм генерации псевдослучайных чисел в режиме счетчика. В этом алгоритме ключ и синхропосылка являются словами фиксированной длины. На шагах алгоритма используются дополнительные входные данные, которые можно выбирать случайным или псевдослучайным методом. Дополнительные входные данные увеличивают неопределенность выходных.

В 6.3 определяется алгоритм генерации псевдослучайных чисел в режиме HMAC. В этом алгоритме ключ и синхропосылка являются словами произвольной длины, дополнительные входные данные не используются. Алгоритм соответствует [2, п. 5].

В приложении А приводится модуль абстрактно-синтаксической нотации версии 1 (АСН.1), определенной в ГОСТ 34.973. Модуль задает идентификаторы алгоритмов и описывает особенности использования в алгоритмах функций хэширования. Рекомендуется использовать модуль при встраивании алгоритмов стандарта в информационные системы, в которых также используется АСН.1.

В приложении Б приводятся примеры выполнения алгоритмов стандарта. Примеры можно использовать для проверки корректности реализаций алгоритмов.

5.2 Функция хэширования

В алгоритмах настоящего стандарта используется функция хэширования h, которая ставит в соответствие сообщению $X \in \{0,1\}^*$ его хэш-значение $h(X) \in \{0,1\}^l$.

Функция h должна быть алгоритмически определена в некотором ТНПА. Например в качестве h могут использоваться функции, заданные в СТБ 1176.1, СТБ 34.101.31. Для обеих этих функций $l = 256$.

В алгоритмах из 6.1, 6.3 требуется, чтобы h была блочно-итерационной. Это значит, что хэширование состоит в итерационной обработке последовательных блоков X, причем блоки являются двоичными словами одной и той же длины b. Функции СТБ 1176.1 и СТБ 34.101.31 являются блочно-итерационными. Для них $b = 256$.

5.3 Ключ

Ключ, который используется при генерации псевдослучайных чисел, должен вырабатываться без возможности предсказания, распространяться с соблюдением мер конфиденциальности и храниться в секрете.

Один и тот же ключ не должен использоваться в различных алгоритмах настоящего стандарта.

Ключом является двоичное слово фиксированной (для алгоритма из 6.2) или произвольной (для алгоритмов из 6.1, 6.3) длины. В алгоритмах из 6.1, 6.3 рекомендуется применять ключ, длина которого совпадает с длиной значений используемой функции хэширования.

Ключ можно генерировать псевдослучайным методом с помощью одного из алгоритмов стандарта. При генерации должен использоваться другой ключ, который также может быть построен псевдослучайным методом еще на одном ключе и т. д. Для построения первоначальных ключей должны применяться генераторы случайных чисел. В приложении В даются справочные сведения по использованию таких генераторов.

5.4 Синхропосылка

Синхропосылка, которая используется при генерации псевдослучайных чисел, должна быть уникальной. Уникальность означает, что при многократном применении алгоритма генерации с одним и тем же ключом вероятность совпадения используемых синхропосылок пренебрежимо мала.

Синхропосылка не является секретным параметром, может быть общедоступной.

Синхропосылки можно вырабатывать случайным или псевдослучайным методом, строить по меткам времени, значениям монотонного счетчика, неповторяющимся номерам сообщений и др.

6 Криптографические алгоритмы генерации псевдослучайных чисел

6.1 Выработка имитовставки по алгоритму HMAC

6.1.1 Функция хэширования

Используется блочно-итерационная функция хэширования h с длиной блока b и значениями длиной l. Должны выполняться ограничения: b кратно 8, $l \leqslant b$.

6.1.2 Входные и выходные данные

Входными данными алгоритма выработки имитовставки являются ключ $\theta \in \{0,1\}^*$ и сообщение $X \in \{0,1\}^*$.

Выходными данными является слово $Y \in \{0,1\}^l$ — имитовставка сообщения X на ключе θ.

6.1.3 Константы и переменные

Слова `ipad`, `opad`. Используются фиксированные слова $\texttt{ipad}, \texttt{opad} \in \{0,1\}^b$, составленные из повторенных $b/8$ раз октетов $\texttt{36}_{16}$ и $\texttt{5C}_{16}$: $\texttt{ipad} = \texttt{36}_{16} \parallel \texttt{36}_{16} \parallel \ldots \parallel \texttt{36}_{16}$, $\texttt{opad} = \texttt{5C}_{16} \parallel \texttt{5C}_{16} \parallel \ldots \parallel \texttt{5C}_{16}$.

Переменная K. Используется переменная K со значениями из $\{0,1\}^b$. Значение K должно быть уничтожено сразу после использования.

6.1.4 Алгоритм HMAC

Вычисление имитовставки сообщения X на ключе θ состоит в выполнении следующих шагов:

1 Если $|\theta| \leqslant b$, то $K \leftarrow \theta \parallel 0^{b-|\theta|}$, иначе $K \leftarrow h(\theta) \parallel 0^{b-l}$.

2 Установить $Y \leftarrow h((K \oplus \texttt{ipad}) \parallel X)$.

3 Установить $Y \leftarrow h((K \oplus \texttt{opad}) \parallel Y)$.

4 Возвратить Y.

6.2 Генерация псевдослучайных чисел в режиме счетчика

6.2.1 Функция хэширования

Используется функция хэширования h со значениями длиной l.

6.2.2 Входные и выходные данные

Входными данными алгоритма генерации псевдослучайных чисел являются натуральное n, ключ $\theta \in \{0,1\}^l$ и синхропосылка $S \in \{0,1\}^l$. Число n определяет количество генерируемых псевдослучайных чисел.

Используется дополнительное входное слово $X \in \{0,1\}^{ln}$. Слово X записывается в виде $X = X_1 \parallel X_2 \parallel \ldots \parallel X_n$, где $X_i \in \{0,1\}^l$ — дополнительные данные, которые используются на i-й итерации алгоритма. Слова X_i могут выбираться произвольным образом, в том числе случайным или псевдослучайным методом. По умолчанию $X_i = 0^l$.

Выходными данными алгоритма является слово $Y \in \{0,1\}^{ln}$ — псевдослучайные числа, полученные на ключе θ при использовании синхропосылки S и дополнительных данных X. Слово Y записывается в виде $Y = Y_1 \parallel Y_2 \parallel \ldots \parallel Y_n$, где $Y_i \in \{0,1\}^l$.

6.2.3 Переменные

Используются переменные s, r со значениями из $\{0,1\}^l$. Значение r должно быть уничтожено сразу после использования.

6.2.4 Алгоритм генерации

Генерация псевдослучайных чисел состоит в выполнении следующих шагов:

1 Установить $s \leftarrow S$.
2 Установить $r \leftarrow S \oplus 1^l$.
3 Для $i = 1, 2, \ldots, n$ выполнить:
  1) $Y_i \leftarrow h(\theta \parallel s \parallel X_i \parallel r)$;
  2) $s \leftarrow s \boxplus \langle 1 \rangle_l$;
  3) $r \leftarrow r \oplus Y_i$.
4 Установить $Y \leftarrow Y_1 \parallel Y_2 \parallel \ldots \parallel Y_n$.
5 Возвратить Y.

6.3 Генерация псевдослучайных чисел в режиме HMAC

6.3.1 Функция хэширования и алгоритм HMAC_h

Используется блочно-итерационная функция хэширования h с длиной блока b и значениями длиной l. Должны выполняться ограничения: b кратно 8, $l \leqslant b$.

Функция h используется косвенно — как композиционный элемент алгоритма HMAC из 6.1. Алгоритм HMAC с функцией h обозначается через HMAC_h.

6.3.2 Входные и выходные данные

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

Выходными данными алгоритма является слово $Y \in \{0,1\}^{ln}$ — псевдослучайные числа, полученные на ключе θ при использовании синхропосылки S. Слово Y записывается в виде $Y = Y_1 \parallel Y_2 \parallel \ldots \parallel Y_n$, где $Y_i \in \{0,1\}^l$.

6.3.3 Переменные

Используется переменная r со значениями из $\{0,1\}^l$.

6.3.4 Алгоритм генерации

Генерация псевдослучайных чисел состоит в выполнении следующих шагов:
1 Установить $r \leftarrow \text{HMAC}_h(\theta, S)$.
2 Для $i = 1, 2, \ldots, n$ выполнить:
  1) $Y_i \leftarrow \text{HMAC}_h(\theta, r \parallel S)$;
  2) $r \leftarrow \text{HMAC}_h(\theta, r)$.
3 Установить $Y \leftarrow Y_1 \parallel Y_2 \parallel \ldots \parallel Y_n$.
4 Возвратить Y.

Приложение А
(рекомендуемое)
Модуль АСН.1

А.1 Идентификаторы

Алгоритмам настоящего стандарта присваиваются следующие идентификаторы:

`hmac-hspec`	алгоритм HMAC (см. 6.1) с функций хэширования, определяемой долговременными параметрами;
`hmac-hbelt`	алгоритм HMAC (см. 6.1) с функцией хэширования СТБ 34.101.31;
`brng-ctr-hspec`	алгоритм генерации псевдослучайных чисел в режиме счетчика (см. 6.2) с функцией хэширования, определяемой долговременными параметрами;
`brng-ctr-hbelt`	алгоритм генерации псевдослучайных чисел в режиме счетчика (см. 6.2) с функцией хэширования СТБ 34.101.31;
`brng-ctr-stb11761`	алгоритм генерации псевдослучайных чисел в режиме счетчика (см. 6.2) с функцией хэширования СТБ 1176.1 и дополнительными уточнениями (см. далее);
`brng-hmac-hspec`	алгоритм генерации псевдослучайных чисел в режиме HMAC (см. 6.3) с функцией хэширования, определяемой долговременными параметрами;
`brng-hmac-hbelt`	алгоритм генерации псевдослучайных чисел в режиме HMAC (см. 6.3) с функцией хэширования СТБ 34.101.31.

Идентификатор алгоритма либо явно определяет используемую функцию хэширования h, либо указывает, что h задается ссылочно, через дополнительные параметры алгоритма.

Для задания h рекомендуется использовать тип `AlgorithmIdentifier` АСН.1, определенный в СТБ 34.101.19 следующим образом:

```
AlgorithmIdentifier ::= SEQUENCE {
  algorithm OBJECT IDENTIFIER,
  parameters ANY DEFINED BY algorithm
}
```

Компонент `algorithm` этого типа должен задавать идентификатор алгоритма хэширования, а компонент `parameters` — параметры данного алгоритма.

При использовании функции хэширования СТБ 1176.1:

– компонент `algorithm` должен принимать значение {1 2 112 0 2 0 1176 1 11};

– компонент `parameters` должен иметь тип `OCTET STRING` и принимать значение $\langle H \rangle_{256}$.

Здесь H — долговременный параметр алгоритма хэширования (неотрицательное целое), описанный в п. 5.1 СТБ 1176.1. Долговременный параметр L, также описанный в п. 5.1 СТБ 1176.1, должен равняться 256.

В алгоритме `brng-ctr-stb11761` используется алгоритм хэширования СТБ 1176.1, в котором на шаге 15 вместо проверки $d = n + 2$ используется проверка $d = n + 1$, а параметр H задается так, что

$$\langle H \rangle_{256} = \texttt{4E4E9C9C 9C9C4E4E 9C9C4E4E 4E4E9C9C 9C9C4E4E 4E4E9C9C 4E4E9C9C 9C9C4E4E}_{16}.$$

При таких уточнениях алгоритм `brng-ctr-stb11761` соответствует алгоритму, определенному в [3].

A.2 Модуль ACH.1

```
Brng-module-v1 {iso(1) member-body(2) by(112) 0 2 0 34 101 47 module(1) ver1(1)}
DEFINITIONS ::=
BEGIN
  brng OBJECT IDENTIFIER ::= {1 2 112 0 2 0 34 101 47}

  hmac-hspec OBJECT IDENTIFIER ::= {brng 11}
  hmac-hbelt OBJECT IDENTIFIER ::= {brng 12}
  brng-ctr-hspec OBJECT IDENTIFIER ::= {brng 21}
  brng-ctr-hbelt OBJECT IDENTIFIER ::= {brng 22}
  brng-ctr-stb11761 OBJECT IDENTIFIER ::= {brng 23}
  brng-hmac-hspec OBJECT IDENTIFIER ::= {brng 31}
  brng-hmac-hbelt OBJECT IDENTIFIER ::= {brng 32}
END
```

Приложение Б
(справочное)
Тестовые примеры

Б.1 Выработка имитовставки по алгоритму HMAC

В таблице Б.1 представлен пример выполнения алгоритма `hmac-hbelt`. Здесь и далее названия алгоритмов даются в соответствии с приложением А.

Таблица Б.1 — Выработка имитовставки (алгоритм `hmac-hbelt`)

θ	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03_{16}
X	BE329713 43FC9A48 A02A885F 194B09A1 7ECDA4D0 1544AF8C A58450BF $66D2E88A_{16}$
Y	D4828E63 12B08BB8 3C9FA653 5A463554 9E411FD1 1C0D8289 359A1130 $E930676B_{16}$
θ	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 $03A98BF6_{16}$
X	BE329713 43FC9A48 A02A885F 194B09A1 7ECDA4D0 1544AF8C A58450BF $66D2E88A_{16}$
Y	41FFE864 5AEC0612 E952D2CD F8DD508F 3E4A1D9B 53F6A1DB 293B19FE $76B1879F_{16}$
θ	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6 92BD9B1C E5D14101 5445_{16}
X	BE329713 43FC9A48 A02A885F 194B09A1 7ECDA4D0 1544AF8C A58450BF $66D2E88A_{16}$
Y	7D01B84D 2315C332 277B3653 D7EC6470 7EBA7CDF F7FF7007 7B1DECBD $68F2A144_{16}$

Б.2 Генерация псевдослучайных чисел в режиме счетчика

В таблицах Б.2, Б.3 представлены примеры генерации псевдослучайных чисел с помощью алгоритмов `brng-ctr-hbelt` и `brng-ctr-stb11761`.

Таблица Б.2 — Генерация псевдослучайных чисел (алгоритм `brng-ctr-hbelt`)

n	3
θ	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 $03A98BF6_{16}$
S	BE329713 43FC9A48 A02A885F 194B09A1 7ECDA4D0 1544AF8C A58450BF $66D2E88A_{16}$
X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B 5CB0C0FF 33C356B8 35C405AE D8E07F99 E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 $D5BB9C4F_{16}$
Y	1F66B5B8 4B733967 4533F032 9C74F218 34281FED 0732429E 0C79235F C273E269 4C0E74B2 CD5811AD 21F23DE7 E0FA742C 3ED6EC48 3C461CE1 5C33A77A A308B7D2 0F51D913 47617C20 BD4AB07A EF4F26A1 AD1362A8 F9A3D42F BE1B8E6F $1C88AAD5_{16}$

Таблица Б.3 — Генерация псевдослучайных чисел (алгоритм `brng-ctr-stb11761`)

n	3
θ	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
S	BE329713 43FC9A48 A02A885F 194B09A1 7ECDA4D0 1544AF8C A58450BF 66D2E88A$_{16}$
X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B 5CB0C0FF 33C356B8 35C405AE D8E07F99 E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F$_{16}$
Y	F7619A01 893ED14E CA0FF583 2797086C FB5B2F90 8CFB4B33 4BC201C9 814D744F 3F616631 B040A16E 0D1EC7A7 CC62000C 377869AD 874E473C 58493143 9FC6D41D 4DAFA372 72A93832 BA8D405F 1DC58F70 B943CAC3 3A926789 C8A3C819 E6F24F98$_{16}$

Б.3 Генерация псевдослучайных чисел в режиме HMAC

В таблице Б.4 представлен пример генерации псевдослучайных чисел с помощью алгоритма `brng-hmac-hbelt`.

Таблица Б.4 — Генерация псевдослучайных чисел (алгоритм `brng-hmac-hbelt`)

n	3
θ	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
S	BE329713 43FC9A48 A02A885F 194B09A1 7ECDA4D0 1544AF8C A58450BF 66D2E88A$_{16}$
Y	AF907A0E 470A3A1B 268ECCCC C0B90F23 9FE94A2D C6E01417 9FC789CB 3C3887E4 695C6B96 B84948F8 D76924E2 2260859D B9B5FE75 7BEDA2E1 7103EE44 655A9FEF 648077CC C5002E05 61C6EF51 2C513B8C 24B4F3A1 57221CFB C1597E96 9778C1E4$_{16}$

Приложение В

(справочное)

Использование генераторов случайных чисел

Генератор случайных чисел вырабатывает последовательности, каждый следующий элемент которых статистически и вычислительно трудно предсказать по всем предыдущим элементам. Генератор использует один или несколько источников случайности и включает средства обработки данных от источников.

В компьютерных системах распространены следующие источники случайности:

– физические источники, использующие процессы в физических устройствах (например, шум в радиоэлектронных приборах);

– системные источники, использующие состояния, процессы и события операционной системы (системное время, сетевая активность, прерывания);

– источники, основанные на активности операторов (движения мышью, нажатия клавиш).

Предпочтительным является использование физических источников случайности.

Для источника случайности S проводится оценка энтропии (неопределенности, вариабельности) его выходных последовательностей. Для этого строится вероятностная модель источника S и в рамках этой модели определяется величина h такая, что основная вероятностная масса выходных последовательностей длины n сосредоточена на множестве мощности 2^{nh}. Величина h называется удельной энтропией на наблюдение. Например, если S выдает случайные независимые символы алфавита A и вероятность появления символа α равняется p_α, то удельная энтропия рассчитывается по формуле

$$h = -\sum_{\alpha \in A} p_\alpha \log_2 p_\alpha \quad (0 \cdot \log_2 0 = 0).$$

Кроме h существуют и другие характеристики неопределенности. Например, в криптографических приложениях используют минимальную удельную энтропию h_{min}, которая характеризует сложность предсказания самой вероятной выходной последовательности S. Для описанного выше источника минимальная удельная энтропия определяется как

$$h_{min} = \min_{\alpha \in A}(-\log_2 p_\alpha).$$

Оценка величины h (или h_{min}) является сложной задачей, если распределение $\{p_\alpha\}$ известно не полностью, источник S не является стационарным, между выходными символами S имеются зависимости и в других ситуациях. Для оценки h могут применяться статистические методы, основанные на частотах встречаемости в выходных последовательностях m-грамм, а также алгоритмические методы, основанные на коэффициентах обратимого или необратимого сжатия выходных последовательностей.

Если удельная энтропия h оценена, то можно сделать вывод о том, что для надежной генерации секретного ключа длины l требуется использовать не менее l/h наблюдений от источника случайности.

Библиография

[1] Krawchuk H., Bellare M., Canetti R. HMAC: Keyed-Hashing for Message Authentication
Request for Comments: 2104, 1997.

[2] Dierks T., Rescorla E. The Transport Layer Security (TLS) Protocol. Version 1.2
Request for Comments: 5246, 2008.

[3] РД РБ 07040.1206-2003. Руководящий документ Республики Беларусь. Автоматизированная система межбанковских расчетов. Процедура выработки псевдослучайных данных с использованием секретного параметра
Мн.: Национальный Банк Республики Беларусь, 2003.