УДК 691.321

Бездетный В.Т.

КЛАССИФИКАЦИЯ МОДЕЛЕЙ УГРОЗ ДЛЯ ГЕНЕРАТОРОВ СЛУЧАЙНЫХ И ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ

Вступление

Генераторы случайных и псевдослучайных чисел имеют широкое применение практически во всех криптографических приложениях.

Генератор случайных чисел (ГСЧ) – это алгоритм, входом которого является случайная последовательность (полученная от физического источника случайности), а выходом – некоторая последовательность, для улучшения статистических свойств которой используется схема выравнивания. Схема выравнивания предназначена для устранения межсимвольных зависимостей и улучшения статистических свойств последовательности.

Генератор псевдослучайных чисел (ГПСЧ) – это алгоритм, который преобразовывает случайную последовательность небольшой длины в более длинную последовательность. При этом, в отличие от ГСЧ, если на вход ГПСЧ поступят одинаковые последовательности, то выходные последовательности тоже будут одинаковыми.

Для определения и формализации атак на генераторы целесообразно пользоваться разными моделями (определениями) генераторов.

Определение генератора как конечного автомата [4]

Определение 1. Генератор (ГСЧ или ГПСЧ) можно определить как конечный автомат $A_г = (X, S, Y, K, z, g, h, f)$, где X – множество дополнительных входов; S – множество состояний; Y – множество выходов; K – ключевое множество; $z : K\ (\varnothing) \rightarrow S$ – функция инициализации; $g : S \times K \times X \rightarrow K$ – функция обновления ключа; $h : S \times K \times X \rightarrow S$ – функция переходов; $f : S \times K \times X \rightarrow Y$ – функция выходов.

Работу генератора можно описать следующим образом:

2. Процедура начальной инициализации генератора:

k_1 – начальное значение ключа, некоторая случайная последовательность;

$s_1 = z(k_1)$ либо s_1 выбирается случайно – инициализация начального состояния;

3. i-й такт работы генератора ($i \geq 1$):

x_i – дополнительный вход (короче – вход);

$y_i = f(s_i, k_i, x_i)$ – выходное значение (короче – выход);

$s_{i+1} = h(s_i, k_i, x_i)$ – переход в следующее состояние;

$k_{i+1} = g(s_i, k_i, x_i)$ – выбор нового тактового ключа.

В зависимости от свойств системы объектов, которые задают автомат, генераторы делятся на ГСЧ и ГПСЧ.

Определение 2. Генератор случайных чисел $A_{гсч}$ – это конечный автомат, у которого дополнительные входы суть случайны, то есть x_i выбирается из множества X в соответствии с некоторым (нетривиальным) вероятностным распределением.

Определение 3. Генератор псевдослучайных чисел $A_{гпсч}$ – это конечный автомат, у которого дополнительные входы выбираются либо по определенному закону (детерминированному), либо мощность множества дополнительных входов X не больше 1.

Если, например, $X = \varnothing$, то автономный конечный автомат $A_{кг} = (S, Y, K, z, g, h, f)$ называется *генератором псевдослучайных чисел без дополнительных входов*.

Определение генератора в терминах теории сложности [5]

Функцией расширения называется функция $l : \mathrm{N} \rightarrow \mathrm{N}$ такая, что для всех n выполняется $l(n) > n$.

Определение 4. Генератор псевдослучайных чисел с расширением l – это полиномиальный детерминированный алгоритм G, который переводит двоичную последовательность $x \in \{0,1\}^n$ в последовательность $G(x) \in \{0,1\}^{l(n)}$.

Последовательности случайных величин $\{\xi_n\}_{n \in \mathrm{N}}$ и $\{\eta_n\}_{n \in \mathrm{N}}$ называются *неразличимыми за полиномиальное время*, если для любого полиномиального вероятностного

алгоритма A и для любого $C>0$ существует $N=N(C)$ такое, что при любом $n>N$ выполняется неравенство $|P(A(\xi_n)=1) - P(A(\eta_n)=1)| < n^{-C}$.

Определение 5. Пусть случайная величина x равномерно распределена на $\{0,1\}^n$, тогда $\zeta_{l(n)} = G(x)$ – случайная величина со значениями в $\{0,1\}^{l(n)}$. Обозначим через $U_{l(n)}$ случайную величину, которая равномерно распределена на $\{0,1\}^{l(n)}$. Если последовательности $\{\zeta_{l(n)}\}_{n \in N}$ и $\{U_{l(n)}\}_{n \in N}$ неразличимы за полиномиальное время, то генератор G называется *криптографически надежным генератором псевдослучайных чисел.*

Говорят, что ГПСЧ *проходит все полиномиальные статистические тесты*, если никакой полиномиальный алгоритм не отличит выходную последовательность от случайной с вероятностью, существенно большей 1/2. Говорят, что ГПСЧ *проходит тест «следующего бита»*, если никакой полиномиальный алгоритм, получающий на вход первые $n-1$ бит выходной последовательности генератора, не предскажет n-й бит с вероятностью, существенно отличной от 1/2.

Следующие утверждения эквивалентны [3, 5, 6]:

- генератор псевдослучайных чисел крипографически надежен;
- генератор псевдослучайных чисел проходит тест «следующего бита»;
- генератор псевдослучайных чисел проходит все полиномиальные статистические тесты.

Исходя из этого на практике ГПСЧ считают хорошим (непредсказуемым), если выходная последовательность генератора проходит все тесты на случайность из некоторого выбранного набора тестов – таким образом обосновывается неразличимость выходной последовательности генератора от истинно случайной.

Атаки на генераторы, их классификация

Определение 6. Атака на генератор – это попытка противника получить информацию о секретном ключе преобразований, используемых генератором, о внутренних состояниях генератора, о дополнительных входах генератора, о выходной последовательности.

Определение 7. Полное вскрытие – это удачно реализованная атака, в результате которой противнику становятся известны секретные параметры генератора, с помощью которых восстанавливается вся выходная последовательность генератора.

Определение 8. Частичное вскрытие – это удачно реализованная атака, в результате которой противник узнает часть информации о генераторе и может получить (или же с большой вероятностью предсказать) часть выходной последовательности.

Атаку можно характеризовать вероятностью успеха ее проведения, средним (максимальным) временем и объемом памяти, необходимым для ее реализации, объемом дополнительной необходимой информации (биты ключа, дополнительные входы и т.п.).

Далее будем полагать, что противнику известна схема работы генератора, то есть все функции z, g, h, f и множества (структура множеств) X, S, Y, K, а не известны конкретные значения переменных y_i, s_i, k_i, x_i, если не оговаривается противное.

Рассмотрим следующую классификацию атак на генератор [1]:

1.Прямые криптоаналитические атаки.

Предполагается, что противнику известна часть выходной последовательности генератора $\{y_{t+1}, y_{t+2}, y_{t+3}, \ldots, y_{t+n}\}$. Это может произойти, если он может отличить за полиномиальное время выходную последовательность генератора от случайной последовательности или же у него есть непосредственный доступ к выходной последовательности. На основании этой информации противник пытается восстановить неизвестные значения выходной последовательности, определить состояния, ключи, дополнительные входы генератора с целью восстановления выходной последовательности:

1.1. Атака на выход «чтение назад» – противник пытается определить предыдущие значения выходной последовательности по известным элементам выходной последовательности;

1.2. Атака на выход «чтение вперед» – противник пытается предсказать значения выходной последовательности по известным элементам выходной последовательности.

Примечание. Данной атаке противостоят криптографически надежные ГПСЧ;

1.3. Атаки, связанные с периодичностью выходной последовательности: противнику известно о наличии периода (статистического периода) в выходной последовательности, он пытается определить неизвестные значения в выходной последовательности.

2. Атаки по входу.

Предполагается, что противник имеет доступ к информации, которая используется в качестве дополнительного входа генератора. С ее помощью он пытается определить состояния, ключи, выходную последовательность:

2.1. Атака по известному входу – противнику известны значения (или часть значений) x_i;

2.2. Атака по выбранному входу – противник может выбирать (изменять) значения x_i;

2.3. Атака по повторному входу – противнику известно, что некоторая последовательность $\{x_i, x_{i+1}, x_{i+2}, ..., x_{i+t}\}$ использовалась в качестве дополнительного входа повторно.

3. Атаки распространения компрометации состояний.

Предполагается, что были скомпрометированы (стали известны противнику) некоторые состояния генератора. Противник пытается определить неизвестные ему состояния, ключи, выходную последовательность:

3.1. Атака на состояния «чтение назад» – противнику известно состояние $s_i = h(s_{i-1}, k_{i-1}, x_{i-1})$ и он пытается восстановить состояние s_{i-1};

3.2. Атака постоянной компрометации – противнику известно состояние s_i и, используя это, он пытается определить все предыдущие и последующие состояния;

3.3. Атака итерактивного угадывания состояния – противнику известно состояние s_i и он пытается определить состояние s_{i+t};

3.4. Атака на состояния «встреча посредине» – попытка восстановления состояния s_{i+t} генератора при компрометации состояний s_i и s_{i+2t}.

Анализ различных генераторов

Генератор Blum-Blum-Shub (BBS)

$N = p \cdot q$, где p, q – простые числа, такие, что $p = 3 \bmod 4$; $q = 3 \bmod 4$. Значение n (число Блюма) может быть общедоступным, p, q должны сохраняться в секрете.

1. Инициализация генератора:

k – случайное натуральное число меньшее n, взаимно простое с n;

$s_1 = k^2 \bmod n$ – начальное состояние генератора.

2. i-й такт работы генератора ($i \geq 1$):

$y_i = f(s_i) = (s_i^2 \bmod n) \bmod 2$ – выходной бит;

$s_{i+1} = h(s_i) = s_i^2 \bmod n$ – переход в следующее состояние.

Прямые криптоаналитические атаки. Генератор BBS непредсказуем влево и непредсказуем вправо [7]: таким образом, он устойчив к атакам на выход «чтение вперед» и «чтение назад», хотя есть возможность возникновения последовательности малого периода, что делает выходную последовательность предсказуемой.

Генератор BBS не имеет дополнительных входов, поэтому атаки по входу не рассматриваются.

Атаки распространения компрометации состояний. Компрометация любого состояния s_i влечет за собой компрометацию всех последующих состояний: при этом становится известной вся выходная последовательность начиная с y_i. Квадратное уравнение $s_{i+1} = s_i^2 \bmod n$ имеет четыре корня (один из которых тоже является квадратом), но найти их легко, если известны числа p и q, иначе нахождение корней квадратного уравнения равносильно разложению n на множители. Таким образом, генератор BBS устойчив к атаке на состояния «чтение назад».

Генератор, определенный в приложении А к ДСТУ 4145-2002

Описание генератора.

$E_k(\cdot)$ – алгоритм шифрования, определенный ГОСТ 28147-89 в режиме простой замены, на ключе k длины 256 бит;

s, D, I, x – 64 битные строки.

Установка начального состояния.

Начальное состояние s генератора получают от физического источника случайности;

D – текущее значение даты и времени с точностью 64 двоичных разрядов;

$I = E_k(D)$.

При каждом обращении к генератору выполняются следующие вычисления:

$x = E_k(I \oplus s)$;

$s = E_k(x \oplus I)$;

выходным битом есть крайний правый бит числа x, то есть $y = x \bmod 2$.

Описание генератора в терминах определения 1.

$E_k(\cdot)$ – алгоритм шифрования, определенный ГОСТ 28147-89 в режиме простой замены, на ключе k длины 256 бит.

1. Процедура начальной инициализации генератора:

k – секретный ключ криптографического преобразования $E_k(\cdot)$, некоторая последовательность длины 256 бит;

s_1 – начальное состояние, 64 бита, полученные от физического источника случайности;

D – текущее значение даты и времени с точностью 64 бит.

$X = I = E_k(D)$ – дополнительный вход генератора, принимает постоянное значение.

2. i-й такт работы генератора ($i \geq 1$):

$y_i = f(s_i, k, x) = E_k(x \oplus s_i) \bmod 2$ – выходной бит;

$s_{i+1} = h(s_i, k, x) = E_k(E_k(x \oplus s_i) \oplus x)$ – переход в следующее состояние.

Прямая криптоаналитическая атака. Если считать, что $E_k(\cdot)$ односторонняя функция, а функция, ставящая в соответствие двоичному вектору его крайний правый бит, есть ядро функции $E_k(\cdot)$, то рассматриваемый генератор криптографически надежен, а значит, противостоит атаке на выход «чтение вперед». Аналогично сказанному, если считать расшифрование $E_k^*(\cdot)$ односторонней функцией, то генератор противостоит атаке на выход «чтение назад».

Атаки по входу. Даже если противник может контролировать значение D, без знания секретного ключа k он не сможет ослабить генератор.

Атаки распространения компрометации состояний. Предположим, что скомпрометирован ключ k. Противник может реализовать атаку на состояния «встреча посредине» для определения состояния s_i по скомпрометированным соседним состояниям s_{i-1} и s_{i+1} следующим образом: с одной стороны – $s_i = E_k(E_k(E_k(D) \oplus s_{i-1}) \oplus E_k(D))$, а с другой – $s_i = E_k^*(E_k^*(s_{i+1}) \oplus E_k(D)) \oplus E_k(D)$. Исходя из того, что метка времени D обладает небольшой энтропией, противник может подобрать необходимое значение D.

Генератор ANSI X9.17

Рассмотрим в качестве блочного алгоритма шифрования $E_k(\cdot)$, который используется в ANSI X9.17, алгоритм шифрования, определенный ГОСТ 28147-89 в режиме простой замены, на ключе k длины 256 бит.

1. Процедура начальной инициализации генератора:

k – секретный ключ криптографического преобразования $E_k(\cdot)$, некоторая последовательность длины 256 бит;

s_1 – начальное состояние, 64 бита, полученные от физического источника случайности;

2. i-й такт работы генератора ($i \geq 1$):

x_i – текущая метка времени длиной 64 бита, дополнительный вход;

$y_i = f(s_i, k_i, x_i) = E_k(E_k(x_i) \oplus s_i)$ – выходной вектор, 64 бита;

$s_{i+1} = h(s_i, k_i, x_i) = E_k(E_k(x_i) \oplus y_i) = E_k(E_k(x_i) \oplus E_k(E_k(x_i) \oplus s_i))$ – переход в следующее состояние.

Прямые криптоаналитические атаки сводятся к криптоанализу ГОСТ 28147-89.

Атаки по входу. Теоретически реализуема атака по повторному входу. Если противник сможет зафиксировать метку времени x_i = const, то он сможет отличить выходную последовательность ANSI X9.17 от случайной, имея около 2^{32} 64-битных выходных символов. Поскольку в последовательности случайных 64-битных чисел коллизия будет наблюдаться после получения приблизительно 2^{32} выходов, а при фиксированном входе ANSI X9.17 для получения коллизии необходимо примерно 2^{63} выходов. Но тем не менее знание или контроль входов не ослабит генератора, если секретный ключ неизвестен.

Атаки распространения компрометации состояний. Предположим, что скомпрометирован ключ k. Если противнику известны два последовательных выходных значения y_i, y_{i+1}, то он сможет скомпрометировать состояние s_{i+1} следующим образом:

$$s_{i+1} = E_k(E_k(x_i) \oplus y_i) \text{ и } s_{i+1} = E_k^*(y_{i+1}) \oplus E_k(x_{i+1}),$$

подобрать необходимые значения x_i и x_{i+1}, поскольку метки времени имеют малую энтропию.

Противник может реализовать атаку на состояния «встреча посредине» для определения состояния s_i по скомпрометированным соседним состояниям s_{i-1} и s_{i+1}. С одной стороны, $s_i = E_k(E_k(x_{i-1}) \oplus E_k(E_k(x_{i-1}) \oplus s_{i-1}))$, с другой – $s_i = E_k^*(E_k^*(s_{i+1}) \oplus E_k(x_i)) \oplus E_k(x_i)$. Исходя из того, что метки времени x_{i-1} и x_i обладают небольшой энтропией, противник может подобрать необходимое значение.

Выводы

В первую очередь, при построении генератора целесообразно использовать криптографические примитивы, имеющие доказуемую стойкость. Однако из этого еще не следует доказуемая стойкость генератора, поскольку при его построении используются дополнительные параметры и функции. Поэтому необходимо проводить оценку стойкости самого генератора, рассматривая ситуации, когда дополнительные параметры являются частично известными.

Следует обеспечить непредсказуемость начального заполнения и избегать использования «слабых» начальных заполнений, которые приводят к ухудшению свойств выходной последовательности.

Перед использованием генератор должен быть протестирован, поскольку положительные результаты тестирования есть необходимым условием его криптографической стойкости.

Значения дополнительных входов должны изменяться недетерминированно, случайно. Например, если в качестве дополнительных входов используются метки времени, то интервалы между ними должны быть случайными.

При построении рассмотренной классификации атак на генераторы учитывались возможности компрометации элементов каждого из множеств, входящих в определение 1, кроме множества ключей. Однако остаются открытыми вопросы о генераторах с изменяющимися ключами и компрометации шаговых ключей.

Список литературы

1. *Kelsey J., Schneier B., Wagner D., Hall C.* Cryptanalytic Attacks on Pseudorandom Number Generators // FSE'98, LNCS 1372, p. 168-188, 1998.

2. *Desai A., Hevia A., Yin Y. L.* A Practice-Oriented Treatment of Pseudorandom Number Generators // EUROCRYPT 2002, LNCS 2332, p. 368-385, 2002.

3. *Goldreich O.* Pseudorandomness // Notices of the AMS, vol. 46, 10, pp. 1209-1216, 1999.

4. *Фомичев В. М.* Дискретная математика и криптология. – М., 2003, с. 260-269.

5. *Вербицький О. В.* Основи криптології. -Львів, ВНТЛ, 1998, с. 152-161.

6. *Menezes A., Van Oorschot P., Vanstone S.* Handbook of Applied Cryptography //CRC Press, 1996.

7. *Шнайер Б.* Прикладная криптография // ТРИУМФ. - Москва, 2002.

Поступила 14.03.2006