EMERCOIN

РусКрипто-2024

Re:Crypt

RC4OK. МОДИФИЦИРОВАННАЯ ВЕРСИЯ RC4

Олег Игоревич Ховайко, «Эмеркоин», olegh@emercoin.com

к.т.н., Дмитрий Анатольевич Щелкунов, «Рекрипт», schelkunov@re-crypt.com

Генераторы псевдослучайных чисел

- ✓ Широко используются для работы разнообразного ПО
- ✓ Широко используются в IoT
- ✓ Зачастую встроены в ОС
- ✓ **Для добавления внешней энтропии используют механизмы синхронизации, что сопряжено с накладными расходами**

ГПСЧ на базе RC4

- ✓ Высокая скорость
- ✓ Очень простая реализация
- ✓ ГПСЧ на базе RC4 является потоковым - позволяет генерировать последовательности байт любой длины
- ✓ **RC4 логично использовать в качестве базового алгоритма при реализации ГПСЧ**

Классический RC4. KSA

$i \leftarrow 0$
while $i \leq 255$ **do**
    $S[i] \leftarrow i$
    $i \leftarrow i + 1$
end while
$j \leftarrow 0$
$i \leftarrow 0$
while $i \leq 255$ **do**
    $j \leftarrow (j + S[i] + key[i \mod keylength]) \mod 256$
    $t \leftarrow S[i]$
    $S[i] \leftarrow S[j]$
    $S[j] \leftarrow t$
    $i \leftarrow i + 1$
end while

Классический RC4. PRGA

$i \leftarrow 0$
$j \leftarrow 0$
while $GeneratingOutput$ **do**
    $i \leftarrow (i + 1) \mod 256$
    $j \leftarrow (j + S[i]) \mod 256$
    $t \leftarrow S[i]$
    $S[i] \leftarrow S[j]$
    $S[j] \leftarrow t$
    $u \leftarrow (S[i] + S[j]) \mod 256$
    $K \leftarrow S[u]$ ▷ Output K
end while

Атаки на RC4

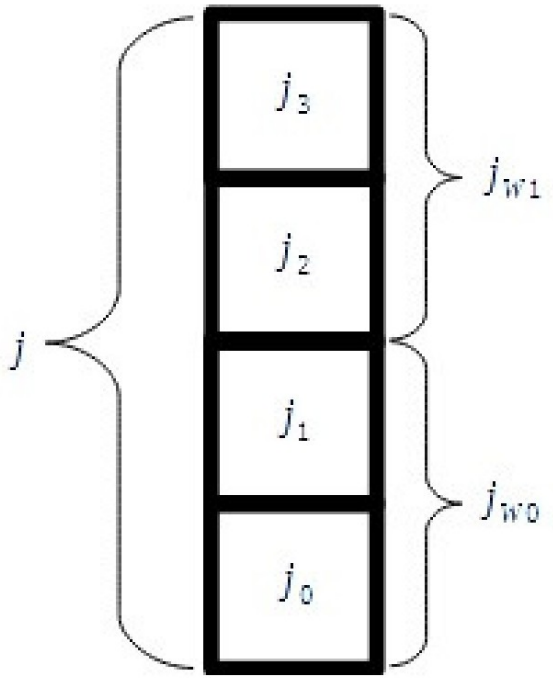
- ✓John Leyden, That earth-shattering NSA crypto-cracking: Have spooks smashed RC4?, 2013, https://www.theregister.com/2013/09/06/nsa_cryptobreaking_bullrun_analysis/
- ✓Green, Matthew, Attack of the week: RC4 is kind of broken in TLS, 2013, https://blog.cryptographyengineering.com/2013/03/12/attack-of-week-rc4-is-kind-of-broken-in/
- ✓Andrei Popov, Prohibiting RC4 Cipher Suites, 2015, doi:10.17487/RFC7465
- ✓Sepehrdad, P., Vaudenay, S., Vuagnoux, M., Discovery and Exploitation of New Biases in RC4, 2011
- ✓Mozilla Security Server Side TLS Recommended Configurations, 2015, https://wiki.mozilla.org/Security/Server_Side_TLS
- ✓Security Advisory 2868725: Recommendation to disable RC4, 2013, http://blogs.technet.com/b/srd/archive/2013/11/12/security-advisory-2868725-recommendation-to-disable-rc4.aspx

Атаки эксплуатируют «слабости» KSA

RC4OK. Модифицированный RC4

- ✓По совету автора оригинального RC4, Рона Райвеста, алгоритм был назван RC4OK, где суффикс OK – инициалы первого автора
- ✓Изменён алгоритм PRGA
- ✓Изменён алгоритм KSA
- ✓«Отбрасываются» первые 256 байт, сгенерированные ГПСЧ на базе RC4OK
- ✓**Создан алгоритм добавления энтропии из внешнего истточника без использования дополнительных механизмов синхронизации**

RC4OK. Модифицированный RC4. PRGA



$i \leftarrow S[j \oplus 85]$ ▷ Randomize i
$j \leftarrow 0$ ▷ Clear j
while *GeneratingOutput* **do**
  $i \leftarrow (i + 11) \mod 256$
  $j \leftarrow (j << 1) + (j >> 31)$ ▷ circular left shift by 1
  $j_0 \leftarrow (j_0 + S[i]) \mod 256$
  $t \leftarrow S[i]$
  $S[i] \leftarrow S[j_0]$
  $S[j_0] \leftarrow t$
  $u \leftarrow (S[i] + S[j_0]) \mod 256$
  $K \leftarrow S[u]$ ▷ Output K
end while

RC4OK. Модифицированный RC4. PRGA

$i \leftarrow S[j \oplus 85]$ ▷ Randomize i

Рандомизация не ухудшает свойств по сравнению с оригинальным RC4

$j \leftarrow 0$ ▷ Clear j

while *GeneratingOutput* **do**

$i \leftarrow (i + 11) \mod 256$

Модификация не ухудшает свойств по сравнению с оригинальным RC4

$j \leftarrow (j << 1) + (j >> 31)$

Необходимо для добавления энтропии из внешнего источника и для воспрепятствования входу ГПСЧ в возможное редуцированное кольцо состояний, отвечающее, в частности, за "плохие" результаты PractRand

$j_0 \leftarrow (j_0 + S[i]) \mod 256$

$t \leftarrow S[i]$

$S[i] \leftarrow S[j_0]$

$S[j_0] \leftarrow t$

$u \leftarrow (S[i] + S[j_0]) \mod 256$

$K \leftarrow S[u]$ ▷ Output K

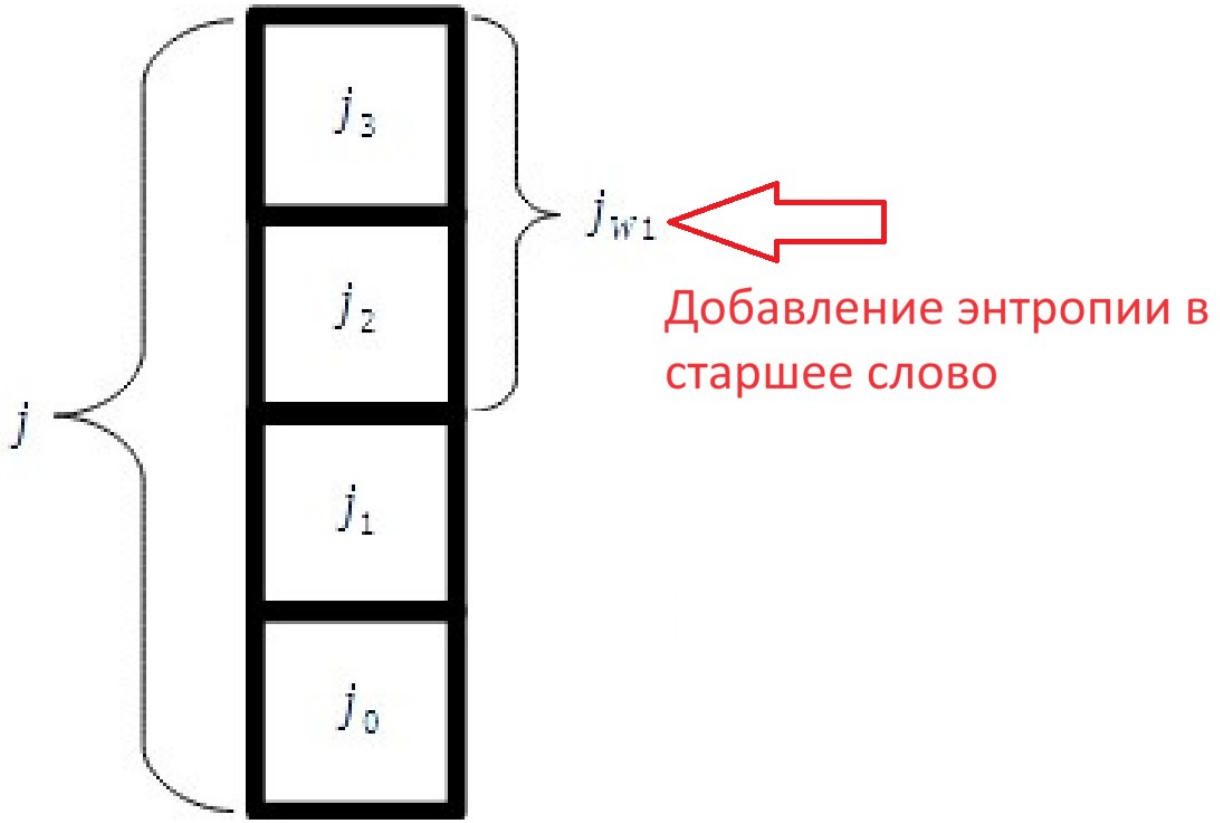
end while

RC4OK. Модифицированный RC4. KSA

$i \leftarrow 0$
$j \leftarrow 0$
while $i \leq 255$ **do**
    $j \leftarrow j + 233 \mod 256$ ⇦ Не ухудшает оригинальный KSA
    $S[i] \leftarrow j$
    $i \leftarrow i + 1$
end while
$j \leftarrow 0$
$i \leftarrow 0$
while $i \leq 255$ **do**
    $j \leftarrow (j + S[i] + key[i \mod keylength]) \mod 256$
    $t \leftarrow S[i]$
    $S[i] \leftarrow S[j]$
    $S[j] \leftarrow t$
    $i \leftarrow i + 1$
end while

После того как KSA отработает, необходимо извлечь из генератора не менее чем 256 байт и проигнорировать их

RC4OK. Модифицированный RC4. Добавление энтропии



while $HarvestingEntropy$ **do**
$j_{w1} \leftarrow (j_{w1} << 1) + (j_{w1} >> 15)$
$j_{w1} \leftarrow j_{w1} + entropy \mod 65536$
end while

RC4OK. Модифицированный RC4. Добавление энтропии. Коллизии

- ✓ Память (32-х битная переменная j) разделяется между двумя потоками, поэтому возможны коллизии при доступе к этой памяти
- ✓ Однако при детальном рассмотрении очевидно, что эпизодически возникающие коллизии, напротив, добавляют неопределённости в логику работы ГПСЧ, что является конечной целью механизма добавления энтропии

RC4OK. Показатели

- ✓Скорость ГПСЧ около 300 Мбайт\сек (процессор Intel Core i7)
- ✓Требуемая ОЗУ: 260 байт
- ✓Тесты PractRand показали хорошие результаты более чем на 32 Тб данных (как с внешим источником энтропии, так и без него)
- ✓Тесты PractRand для оригинального RC4 показывают отрицательные результаты на менее чем 1 Тб данных (последовательность признаётся неслучайной)
- ✓После внедрения ГПСЧ на базе RC4OK скорость скачивания блокчейна Emercoin выросла в 2 раза

RC4OK. Ссылки

- ✓https://github.com/emercoin/emercoin/
- ✓https://github.com/emercoin/rc4ok/tree/main
- ✓https://github.com/emercoin/rc4ok/tree/main/PractRandTest
- ✓https://lib.rs/crates/rc4ok
- ✓https://t.me/recryptor