Щелкунов Д.А., к.т.н.
КФ МГТУ имени Н.Э. Баумана

White-Box криптография[1] и SPN-шифры[2]. LRC-метод.

Рассматривается подход, позволяющий скрыть линейную зависимость между элементами T-box таблиц[3] в SPN-шифре (LRC-метод). Приведена методика создания White-Box схем на базе SPN-шифра. Выявлены условия, которым должен соответствовать SPN-шифр, для создания его стойкой White-Box реализации.

Текущее развитие криптографии характеризуется активными исследованиями в области *White-Box* криптографии [1]-[8]. Методы *White-Box* криптографии позволят модифицировать симметричный шифр таким образом, что его можно будет использовать для ассиметричной криптографии. Действительно, если восстановление ключа из *White-Box* реализации алгоритма шифрования будет трудной задачей, также как и создание алгоритма расшифрования по имеющемуся алгоритму шифрования, то пара алгоритмов (алгоритм шифрования – алгоритм расшифрования) становится фактически ключевой парой, где *White-Box* реализация алгоритма шифрования является открытой, а реализация алгоритма расшифрования недоступна аналитику. Такая схема может обладать целым рядом преимуществ по сравнению с классическими ассиметричными криптосистемами. Основным преимуществом является скорость работы.

За последние годы было представлено несколько *White-Box* реализаций известных алгоритмов шифрования. Основной особенностью этих реализаций является встраивание раундовых ключей в таблицы подстановок (*S-box* таблицы). Таким образом, получается, что ни сам ключ шифрования – ни раундовые ключи явным образом в программе не фигурируют. Однако, все эти реализации не являются стойкими по отношению к атакам на основе подобранного открытого текста [6]-[8]. Более того, даже если достаточно трудно восстановить сам ключ, то существует возможность построить таблицы обратных подстановок, а следовательно, возможность построить алгоритм расшифрования по алгоритму шифрования.

В данной работе рассматривается *SPN*-шифр, имеющий структуру, схожую с *Rijndael* [9]. Это связано в первую очередь с тем, что изначальной задачей автора работы являлась разработка модифицированного варианта алгоритма шифрования *AES-128,* позволяющего работать с модифицированным ключом. Целью являлось противодействие атакам, основанным на перехвате сеансового ключа. Задача была решена при помощи подхода, описанного ниже. Однако было выявлено, что если атакующий получает доступ к реализации алгоритма, то у него появляется возможность определить преобразования, используемые для модификации сеансового ключа и, как следствие, создать обратные. Таким образом, был сделан вывод о невозможности применения разработанного подхода для создания стойких *White-Box* схем на базе стандартного алгоритма *Rijndael*.

Для дальнейших рассуждений рассмотрим следующую систему:

$$\begin{cases} x_1 = ((a \cdot b)(mod\ p_1) \cdot c)(mod\ p_2) \\ x_2 = (a \cdot (b \cdot c)(mod\ p_2))(mod\ p_1) \end{cases} \qquad (1)$$

[1] Раздел криптографии, изучающий в частности такие реализации симметричных шифров, в которых ключ шифрования скрыт в самой реализации [1]-[8]. Такие реализации называются *White-Box* реализациями или *White-Box* схемами.
[2] Шифры, основанные на базе подстановочно-перестановочных сетей. Пример такого шифра – *Rijndael* [9].
[3] Таблицы подстановки, в которых каждому входному байту ставится в соответствие последовательность выходных. принципе формирования таких таблиц в шифре *Rijndael* можно почитать в спецификации [9].

Здесь p_1, p_2 – неприводимые полиномы n-й степени, а a,b,c,x_1,x_2 - полиномы степеней, меньших n. Можно предположить, что в общем случае $x_1 \neq x_2$.

Утверждение 1.

В системе (1) найдутся такие полиномы a,b,c, что $x_1 \neq x_2$.

Доказательство:

Если $x_1 = x_2$, то справедливо следующее равенство:

$$a \cdot b \cdot c - q \cdot c \cdot p_1 - v \cdot p_2 = a \cdot b \cdot c - a \cdot u \cdot p_2 - r \cdot p_1 \qquad (2)$$

Здесь q – частное от деления произведения полиномов $a \cdot b$ на p_1, v - частное от деления $(a \cdot b)(mod\ p_1) \cdot c$ на p_2, u – частное от деления произведения полиномов $b \cdot c$ на p_2, а r - частное от деления $(b \cdot c)(mod\ p_2) \cdot a$ на p_1.

Из формулы (2) следует:

$$\frac{q \cdot c - r}{a \cdot u - v} = \frac{p_2}{p_1} \qquad (3)$$

Из равенства (3) очевидно, что $q \cdot c - r$ должно делиться на p_2, но степень полинома r не может превышать степень полинома p_2. То же самое можно сказать и про степени полиномов q,c,a,u и v. Таким образом, равенство (3) выполняется тогда, когда либо один из полиномов a,b или c является нулевым – либо, когда степень произведения полиномов $a \cdot b \cdot c$ не превышает степень полинома p_1 или p_2, что означает, что. $q = r = u = v = 0$, что не всегда выполнимо. Из вышесказанного следует, что в системе (1) в общем случае $x_1 \neq x_2$, что и требовалось доказать.

Утверждение 1 говорит нам о том, что закон ассоциативности при умножении полиномов в различных, пусть и изоморфных, полях, как показывает нам система (1), в общем случае не соблюдается.

Рассмотрим теперь следующую систему:

$$\begin{cases} y_1(x) = (s(x) \cdot a(mod\ p_1)) \cdot b(mod\ p_2) \\ y_2(x) = (s(x) \cdot c(mod\ p_1)) \cdot d(mod\ p_3) \end{cases} \qquad (4)$$

Здесь p_1, p_2, p_3 - неприводимые попарно неравные полиномы одинаковой степени над $GF(2)$, x, a, b, c, d – произвольные полиномы над $GF(2)$, $s(x)$ - нелинейная функция от x. Пусть p_1, p_2, p_3, a, b, c, d, $s(x)$ неизвестны, а функции $y_1(x)$ и $y_2(x)$ заданы с помощью таблиц подстановки. В этом случае задача нахождения линейной зависимости между элементами $s(x) \cdot a(mod\ p_1)$ и $s(x) \cdot c(mod\ p_1)$ при известных значениях $y_1(x)$ и $y_2(x)$ в поле порядка степени полинома p_1 является сложной. Постольку-поскольку, в соответствии с утверждением 1, закон ассоциативности для каждого из выражений системы (4) не выполняется в общем случае, линейной зависимости между $y_1(x)$ и $y_2(x)$ в поле порядка степени полинома p_1 не существует. Поэтому для нахождения линейной зависимости между $s(x) \cdot a(mod\ p_1)$ и $s(x) \cdot c(mod\ p_1)$ необходимо найти b, d, p_2 и p_3. Очевидно, что сложность такой задачи составляет приблизительно 2^{2n}, где n - степень полинома p_1. Модифицируем систему (4) следующим образом:

$$\begin{cases} y_1(x) = (...(s(x)\cdot a(mod\ p_1))\cdot b^{(0)}(mod\ p_2^{(0)})...)\cdot b^{(k)}(mod\ p_u^{(k)}) \\ y_2(x) = (...(s(x)\cdot c(mod\ p_1))\cdot d^{(0)}(mod\ p_3^{(0)})...)\cdot d^{(k)}(mod\ p_v^{(k)}) \end{cases} \quad (5)$$

Здесь $p_i^{(\alpha)} \neq p_j^{(\alpha)}$. В данном случае сложность восстановления линейной зависимости между $s(x)\cdot a(mod\ p_1)$ и $s(x)\cdot c(mod\ p_1)$ будет составлять $2^{2n(k+1)}$.

Таким образом, можно предположить, что если многократно умножать каждый элемент *T-box* таблицы [9] на произвольно выбранные полиномы по модулю произвольно выбранных полиномов 8-й степени, то восстановление линейной зависимости между элементами *T-box*-ов будет сложной задачей. Формально такое умножение выглядит следующим образом:

$$T'_i[a] = \begin{bmatrix} ((...(t_i^{(0)}(a)\cdot b_i^{(0,0)})(mod\ p_i^{(0,0)})\cdot b_i^{(0,1)}(mod\ p_i^{(0,1)})...)\cdot b_i^{(0,k_0)}(mod\ p_i^{(0,k_0)}) \\ ((...(t_i^{(1)}(a)\cdot b_i^{(1,0)})(mod\ p_i^{(1,0)})\cdot b_i^{(1,1)}(mod\ p_i^{(1,1)})...)\cdot b_i^{(1,k_1)}(mod\ p_i^{(1,k_1)}) \\ ((...(t_i^{(2)}(a)\cdot b_i^{(2,0)})(mod\ p_i^{(2,0)})\cdot b_i^{(2,1)}(mod\ p_i^{(2,1)})...)\cdot b_i^{(2,k_2)}(mod\ p_i^{(2,k_2)}) \\ ((...(t_i^{(3)}(a)\cdot b_i^{(3,0)})(mod\ p_i^{(3,0)})\cdot b_i^{(3,1)}(mod\ p_i^{(3,1)})...)\cdot b_i^{(3,k_3)}(mod\ p_i^{(3,k_3)}) \\ \dots\dots\dots\dots\dots\dots \\ ((...(t_i^{(n)}(a)\cdot b_i^{(n,0)})(mod\ p_i^{(n,0)})\cdot b_i^{(n,1)}(mod\ p_i^{(n,1)})...)\cdot b_i^{(n,k_n)}(mod\ p_i^{(n,k_n)}) \end{bmatrix} \quad (6)$$

Здесь $t_i^{(j)}$ - элемент *T-box*-а до применения запутывающих преобразований, $b_i^{(j,u)}$– случайно выбранный полином в $GF(2^8)$, $p_i^{(j,u)}$– случайно выбранный неприводимый полином 8-й степени над $GF(2)$. При этом должно соблюдаться следующее условие:

$$p_i^{(0,v)} \neq p_i^{(1,v)} \neq ... \neq p_i^{(n,v)} \quad (7)$$

Описанный выше подход назовем *LRC*-методом. На основе модифицированных, как показано в формулах (6) и (7), *T-box*-ов можно создать White-Box реализацию *SPN*-шифра. Для шифра *Rijndael,* например, такая модификация будет выглядеть следующим образом:

$$Y_j = \begin{bmatrix} y_j^{(0)} \\ y_j^{(1)} \\ y_j^{(2)} \\ y_j^{(3)} \end{bmatrix} = \begin{bmatrix} mix_j^{(0)}(t_j^{(0,0)}) \\ mix_j^{(1)}(t_j^{(1,0)}) \\ mix_j^{(2)}(t_j^{(2,0)}) \\ mix_j^{(3)}(t_j^{(3,0)}) \end{bmatrix} \oplus \begin{bmatrix} mix_j^{(0)}(t_j^{(0,1)}) \\ mix_j^{(1)}(t_j^{(1,1)}) \\ mix_j^{(2)}(t_j^{(2,1)}) \\ mix_j^{(3)}(t_j^{(3,1)}) \end{bmatrix} \oplus \begin{bmatrix} mix_j^{(0)}(t_j^{(0,2)}) \\ mix_j^{(1)}(t_j^{(1,2)}) \\ mix_j^{(2)}(t_j^{(2,2)}) \\ mix_j^{(3)}(t_j^{(3,2)}) \end{bmatrix} \oplus \begin{bmatrix} mix_j^{(0)}(t_j^{(0,3)}) \\ mix_j^{(1)}(t_j^{(1,3)}) \\ mix_j^{(2)}(t_j^{(2,3)}) \\ mix_j^{(3)}(t_j^{(3,3)}) \end{bmatrix} \quad (8)$$

Здесь $t_j^{(i,k)}$ - элемент *T-box*-а, $mix_j^{(i)}$- преобразование, примененное по отношению к элементу $t_j^{(i,k)}$.

$$mix_j^{(i)}(t_j^{(i,k)}) = ((...(t_j^{(i,k)}\cdot b_j^{(i,0)})(mod\ p_j^{(i,0)})\cdot b_j^{(i,1)}(mod\ p_j^{(i,1)})...)\cdot b_j^{(i,n)}(mod\ p_j^{(i,n)}) \quad (9)$$

В формуле (11) i - индекс элемента *T-box*-а, k - индекс *T-box*-а, j - порядковый номер 4-х байтовой последовательности, $b_j^{(u,v)}$ - произвольный многочлен в $GF(2^8)$, $p_i^{(u,v)}$– случайно выбранный неприводимый полином 8-й степени над $GF(2)$.

Однако применение данного подхода к шифру *Rijndael*, в котором известен принцип формирования таблиц подстановки и полином, используемый в преобразовании *MixColumns*, не позволит противостоять атаке на основе выбранного открытого текста. Действительно, преобразования $mix_j^{(i)}$, представленные в формуле (8) находятся достаточно легко. Ведь нам известно, что

$$t_j^{(i,k)} = s[\,a\,] \cdot n \oplus key_j^{(i,k)} \qquad , \qquad\qquad (10)$$

где a - байт открытого текста, $s[\,a\,]$ - известное нелинейное преобразование в поле *Rijndael*, $key_j^{(i,k)}$ - часть секретного раундового ключа. Операция умножения в формуле (10) производится в поле *Rijndael*. После применения *mix*-преобразования получим:

$$mix_j^{(i)}(t_j^{(i,k)}) = mix_j^{(i)}(s[\,a\,] \cdot n) \oplus mix_j^{(i)}(key_j^{(i,k)}) \qquad\qquad (11)$$

Возьмем два байта открытого текста a и a'. Тогда из формулы (11) следует:

$$mix_j^{(i)}(t_j^{(i,k)}) \oplus mix_j^{(i)}(t'^{(i,k)}_j) = mix_j^{(i)}(n \cdot (s[\,a\,] \oplus s[\,a'\,])) \qquad\qquad (12)$$

В формуле(14) нам известны n, $s[\,a\,]$, $s[\,a'\,]$. Таким образом, мы можем легко найти преобразование $mix_j^{(i)}$, построив таблицу подстановок размером 2^8. После нахождение всех *mix*-преобразований необходимо применить обратные им и получить в результате элементы $t_j^{(i,k)}$, из которых, зная $s[\,a\,]$, легко получить значения $key_j^{(i)}$, что означает восстановление всех раундовых ключей и, как следствие, секретного ключа.

Если значение n в формуле (12) неизвестно, что означает, что неизвестен многочлен над полем $GF(2^8)$, определяющий преобразование *MixColumns*, то взлом вышеописанной схемы все равно возможен. Рассмотрим соседние *T-box*-ы в формуле (8).

$$\begin{cases} t_j^{(0,0)} = s[\,a\,] \cdot n^{(0)} \oplus key_j^{(0,0)} \\ t_j^{(0,1)} = s[\,a\,] \cdot n^{(1)} \oplus key_j^{(0,1)} \\ t_j^{(0,2)} = s[\,a\,] \cdot n^{(2)} \oplus key_j^{(0,2)} \\ t_j^{(0,3)} = s[\,a\,] \cdot n^{(3)} \oplus key_j^{(0,3)} \end{cases} \qquad\qquad (13)$$

Если предположить $n^{(0)} = \alpha$, то, следуя алгоритму для известных таблиц подстановки и преобразования *MixColumns*, можно найти $mix_j^{(0)}$, а следовательно, $n^{(1)}, n^{(2)}, n^{(3)}$. Если предположение $n^{(0)} = \alpha$ верно, то $n^{(0)}, n^{(1)}, n^{(2)}, n^{(3)}$ - коэффициенты многочлена *MixColumns*, что легко проверить, применив обратные преобразования (*InvMixColumns*) и посмотрев влияние входного байта раунда на выходную последовательность (измениться на выходе должен один байт). Таким образом, очевидно, что сложность взлома при неизвестном многочлене над полем $GF(2^8)$, определяющем преобразование *MixColumns*, составляет 2^8.

В случае, когда преобразование *S-box* неизвестно, но применяется одинаково по отношению к каждому байту входной последовательности, взлом все равно возможен. Рассмотрим более подробно этот случай. Пусть в системе (13) $s[\,a\,]$ неизвестно.В этом случае нам достаточно предположить это значение. Пусть $s[\,a\,] = \beta$, а $s[\,a'\,] = \beta'$. Применим теперь вышеописанный алгоритм, предположив $n^{(0)} = \alpha$. Проверка корректности предположения осуществляется также как в вышеописанном случае при неизвестном $n^{(k)}$. Сложность взлома при этом составляет 2^{24}.

Из всего вышесказанного можно сделать выод, что только в случае, когда таблицы *S-box* генерируются случайным образом и уникальны для каждого байта входной последовательности можно говорить о сложности взлома, превышающей 2^{24}. Совершенно очевидно, что создать стойкую *White-Box* реализацию алгоритма *Rijndael* вышеописанным методом невозможно. Более того, даже если в алгоритме *Rijndael* заменить полином в операции *MixColumns* и сделать таблицы *S-box* случайными для каждого байта входной последовательности, то возможен взлом посредством построения таблиц обратной подстановки размером 154 Гб. Это связано с тем, что преобразование *MixColumns* выполняется модулю полинома $x^4 \oplus 1$.

Таким образом, для создания стойкой *White-Box* реализации в соответствии с вышеописанным подходом необходимо автоматически генерировать *SPN*-шифр, который будет отличаться от классического алгоритма *Rijndael* случайно сгенерированными таблицами *S-box* для каждого байта входной последовательности (операция *SubBytes*) и модифицированной операцией *MixColumns*, в которой преобразование будет выполняться по модулю $x^{16} \oplus 1$, а соответствующий полином над $GF(2^8)$ также должен быть выбран случайным образом с учетом наличия обратного ему в кольце полиномов по модулю $x^{16} \oplus 1$.

Как уже было сказано выше, данный подход можно использовать для создания схем на базе шифров с доказанной стойкостью, работающих с модифицированным сеансовым ключом. Действительно, если модифицировать описанным выше способом раундовые ключи и таблицы подстановок, то возможно выполнять операции шифрования и расшифрования на модифицированном ключе. Результаты этих операций будут совпадать с результатами операций на исходном ключе и немодифицированном алгоритме. Смысл такой схемы заключается в том, что атакующий должен получить доступ к модифицированному симметричному алгоритму, что создает дополнительные сложности.

В результате хотелось бы отметить, что дальнейшее развитие описанного выше подхода позволит создавать принципиально новые ассиметричные конструкции, обеспечивающие высокую скорость работы и создающие дополнительную стойкость по отношению к атакам, подразумевающим выполнение кода на недоверенной платформе.

Использованные материалы:

1. S. Chow, P. Eisen, H. Johnson, P.C. van Oorschot, A White-Box DES Implementation for DRM Applications, 2002.
2. S. Chow, P. Eisen, H. Johnson, P.C. van Oorschot, White Box Cryptography and an AES Implementation, 2002.
3. Julien Bringer, Herve Chabanne, Emmanuelle Dottax, White Box Cryptography: Another Attempt, 2006.
4. Hamilton E. Link, William D. Neumann, Clarifying Obfuscation: Improving the Security of White-Box Encoding.
5. www.softwaresecurity.org - Ivan V. Petrov, Seculab JSC.
6. http://www.cosic.esat.kuleuven.be/publications/thesis-152.pdf - B. Wyseur, "White-Box Cryptography," PhD thesis, Katholieke Universiteit Leuven, B. Preneel (promotor), 169+32 pages, 2009.
7. Щелкунов Д.А. О практическом прприменении White-Box криптографии., // Международная конференция РусКрипто, 2009.
8. Щелкунов Д.А. Разработка методик защиты программ от анализа и модификации на основе запутывания кода и данных, Диссертация на соискание ученой степени кандидата технических наук, Мазин А.В. (научный руководитель), 126+18 стр, 2009.
9. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf - AES specification.