Литература

1. А. Кирьянцев, И. Стефанова. Создание приватного сервиса с использованием приложения CRYP2CHAT. Труды Института системного программирования РАН, Том 27. Выпуск 3. 2015 г. Стр. 279-290.
2. Tor: Overview URL: www.torproject.org/about/overview.html.en
3. Как устроен генератор паролей: www.scritub.com/limba/rusa/194620205.php

А.В. Козачок

О НЕКОТОРЫХ ОПРЕДЕЛЕНИЯХ СТОЙКОСТИ НЕРАЗЛИЧИМОЙ ОБФУСКАЦИИ ПРОГРАММНОГО КОДА

(Академия ФСО России)

Обфускацией программы называется всякое ее преобразование, которое сохраняет вычисляемую программой функцию (эквивалентное преобразование), но при этом придает программе такую форму, что извлечение из текста программы (программного кода) ключевой информации об алгоритмах и структурах данных, реализованных в этой программе, становится трудоемкой задачей [1]. Вопрос о существовании обфускаторов, удовлетворяющих различным формальным определениям понятия стойкости, является основным предметом рассмотрения данной статьи.

В 2001 году впервые строгое математическое определение стойкости обфускации программного кода была предложено Бараком [2]: обфускация считается стойкой, если всякий противник может извлечь из текста обфусцированной программы за разумное (полиномиальное относительно размера входных данных) время не больше информации, чем можно было бы получить, проводя тестовые испытания этой программы как "черного ящика". Рассмотрим основные положения теории Барака.

Определение 1. (Обфускация в модели виртуального "черного ящика"). Пусть $\{C_\ell\}_{\ell\in N}$ – класс булевых схем, тогда обфускатором в модели виртуального "черного ящика" для класса схем $\{C_\ell\}_{\ell\in N}$ называется такая (полиномиальная вероятностная машина Тьюринга) PPT O, которая удовлетворяет следующим требованиям [1]:

– функциональная эквивалентность. Для любой схемы $C\in\{C_\ell\}_{\ell\in N}$ и любого входа x существует пренебрежимо малая функция μ, такая что:

$$Pr[(O(C))(x)\neq C(x)]\le\mu(|C|);$$

– полиномиальные издержки. Существуют полином $poly(\cdot)$ такой, что для любой схемы $C\in\{C_\ell\}_{\ell\in N}$:

$$|O(C)|\le poly(|C|);$$

– свойство виртуального "черного ящика": для любой PPT A (противника) существуют такая PPT S (симулятор) и пренебрежимо малая функция μ такая, что для любой схемы $C\in\{C_\ell\}_{\ell\in N}$:

$$|Pr[A(O(C))=1]-Pr[S^C(1^{|C|})=1]|\le\mu(|C|).$$

В работе Барака также было доказано утверждение о невозможности построения идеального обфускатора для всех классов булевых функций, но данное утверждение не исключает возможности построения обфускатора, стойкого в модели виртуального "черного ящика". Исследования стойкости обфускации также были проведены в работах [3–5]. В 2013 году была предложена новая конструкция [6], так называемая "обфускация неразличимости" (indistinguishability obfuscation - iO).

Определение 2. Обфускатором неразличимости iO называется однородная полиномиальная вероятностная машина Тьюринга для булевых схем класса $\{C_\lambda\}$, где λ – параметр безопасности, если обеспечивается выполнение следующих требований [6]:

– функциональная эквивалентность. Существует пренебрежимо малая функция $\mu(\lambda)$ такая, что для всех $\lambda\in\mathbb{N}$ и всякой схемы $C\in C_\lambda$:

$$\forall x: Pr[iO(C(x))=C(x)]\ge 1-\mu(\lambda);$$

– свойство виртуального "черного ящика" (стойкость). Для любой PPT D (распознавателя), существует пренебрежимо малая функция α такая, что для всех $\lambda\in\mathbb{N}$ и любой пары эквивалентных схем $C_1,C_2\in C_\lambda$, имеющих одинаковый размер, распределения вероятностей случайных величин $iO(C_1)$ и $iO(C_2)$ вычислительно неразличимы, то есть выполняется соотношение:

$$\forall x:\left|Pr[D(iO(\lambda,C_1(x)))=1]-Pr[D(iO(\lambda,C_2(x)))=1]\right|\le\alpha(\lambda).$$

Это означает, что не существует алгоритма распознавания обфускации более эффективного, чем обычное предположение, сделанное на основе анализа входов и выходов обфусцированной программы (функции или обфусцированного набора инструкций).

Неразличимость обфусцированных программ означает, что если две эквивалентные программы (одинакового размера) P_1,P_2 такие, что $\forall x: P_1(x)=P_2(x)$, а iO – это обфускатор неразличимости, который принимает на вход программу $P(x)$ и на выходе генерирует новую программу $iO(P(x))$, то $iO(P_1(x))$ и $iO(P_2(x))$ будут вычислительно неразличимы:

$$\exists P_1(x),P_2(x).\forall x: P_1(x)=P_2(x)\to iO(P_1(x))\approx iO(P_2(x)).$$

На основе данной конструкции можно построить доказательство о том, что возможно построение неразличимого обфускатора для всех классов булевых функций, стойкого в модели виртуального "черного ящика". Рассмотрим основу построения обфускатора неразличимости [7]. Обфускатор iO принимает в качестве входных данных программу f, представленную в виде булевой функции, и на выходе генерирует обфусцированную программу $iO(f)$. Фор-

мально, обфускатор неразличимости iO можно считать компилятором, принимающим на вход булеву функцию $f(x)$ и генерирующим на выходе обфусцированную программу $f'(x)=iO(f(x))$. При этом должно выполняться следующее условие: $\forall x: f'(x)=f(x)$. Определение неразличимой обфускации предъявляет следующее требование по стойкости: для любой PPT A (противника) существует вычислительно неограниченный симулятор S, такой, что для каждой схемы C, за полиномиальное время распознаватель D не может различить результат, вычисленный противником A (A принимает обфусцированную программу $O(C)$ в качестве входа), от результата, вычисленного симулятором S, имеющим лишь оракульный доступ к C, при этом S может сделать неограниченное количество запросов к C. Определение стойкости обфускации в модели виртуального "черного ящика" требует, чтобы для любой PPT A (противника) существовал полиномиальный симулятор S такой, что для каждой схемы C, за полиномиальное время распознаватель D не может различить результат, вычисленный противником A (A принимает обфусцированную программу $O(C)$ в качестве входа), от результата, вычисленного симулятором S, имеющим лишь оракульный доступ к C, при этом S может сделать полиномиальное количество запросов к C. В этом смысле определение стойкости неразличимой обфускации является более строгим.

При этом предполагается что, у противника есть доступ к некоторому оракулу $\Re$, реализующему операции в системе дифференциального кодирования. Одним из возможных вариантов доказательства стойкости криптографических преобразований является применение модели случайного оракула [7], под оракулом понимается вспомогательный алгоритм, который выполняется за один шаг работы основного алгоритма. Другими словами, при оценке трудоемкости атак противника время работы оракула не учитывается, учитывается только число обращений к нему. Входные данные, которые передаются оракулу, называются запросом, выходные – ответом. При каждом новом запросе значение функции на данном аргументе выбирается случайным образом. При этом оракул запоминает пару (аргумент, значение) и при повторном запросе для этого аргумента, независимо от того, кто из участников его выдал, будет возвращено все то же запомненное значение. Противник может задавать оракулам любые запросы, получать и анализировать ответы.

Согласно работам [7,8,9] процедура неразличимой обфускации включает в себя пять основных этапов.

1. Преобразование булевой функций в вид ветвящейся программы.
2. Преобразование ветвящейся программы в вид матричной ветвящейся программы.
3. Рандомизация матричной ветвящейся программы.
4. Кодирование рандомизированной матричной ветвящейся программы с помощью схемы дифференциального кодирования (Graded Encoding Scheme).
5. Вычисление обфусцированной функции.

Предложенная схема базируется на понятии полилинейных отображений. В настоящие время известны два наиболее общих подхода к построению полилинейных отображений, реализующих схему дифференциального кодирования. Первый из них [10] базируется на полилинейных отображениях на основе решеток, а второй [11] на целых числах.

Пусть множество U – универсум, $S \subseteq U$ – множество элементов универсума, Z_p – кольцо вычетов по модулю большого простого числа p. Для любого элемента $x \in Z_p$ обозначим $[x]_S$ – разрешенное кодирование x по множеству индексов S. Система дифференциального кодирования, реализующая полилинейное отображение, обеспечивает выполнение следующих основных операций:

– сложение:

$MM.Add([x]_S,[y]_S) \to [x+y]_S$;

– умножение:

$$MM.Mult([x]_{S_1},[y]_{S_2}) \to \begin{cases} [xy]_{S_1 \cup S_2}, \text{если } S_1 \cap S_2 = \varnothing, (S_1 \cup S_2) \subseteq U \\ \perp \end{cases};$$

– проверка кодирования нуля:

$$MM.ZeroTest([x]_S) \to \begin{cases} 1, \text{если } S = U, x = 0 \in Z_p \\ 0 \end{cases}.$$

С точки зрения стойкости обфускации под противником или распознавателем подразумеваем вероятностный алгоритм A, имеющий доступ к вышеописанному оракулу $\Re$. Работа алгоритма заканчивается формированием искомого запроса $ZeroTest(x)$, если, в случае обращения к оракулу $\Re$ с этим запросом, будет получен положительный ответ. Пусть $Adv(A)$ – вероятность того, что будет получен ответ единица. При обосновании стойкости требуется доказать, что для любого противника, имеющего доступ к оракулу в течение полиномиального времени вероятность $Adv(A)$ пренебрежимо мала. Для этого доказывают, что алгоритм A, который решает задачу построения разрешенного полинома $p(\cdot)$, может быть преобразован в алгоритм, который решает другую задачу T. В таких случаях говорят, что задача A сводится к задаче T. Сводимость означает, что если T – трудная задача, то задача построения разрешенного полинома $p(\cdot)$ также трудна.

Существуют различные подходы к доказательству стойкости обфускации, построенные на основе конструкции в работе [7], удовлетворяющие свойству виртуального "черного ящика". В работе [12] была доказана стойкость обфускации, которая заключается в том, что задача деобфускации была сведена к задаче пропозициональной выполнимости булевых формул (SAT – Boolean satisfiability problem). Задача SAT заключается в следующем: как осуществить выбор значений переменных булевой функции таким образом, чтобы значение функции было равно единице. Известно, что задача SAT является NP-полной, а

значит, в общем случае ее решение является вычислительно трудной задачей. Данный подход приемлем в случае корректного доказательства требования неразличимости (свойства виртуального "черного ящика"), а затем сведения задачи построения разрешенного полинома $p(\cdot)$ к задаче SAT.

Таким образом, в статье представлены различные подходы к определению стойкости обфускации и установлена возможность построения универсального обфускатора, удовлетворяющего требованиям стойкости в модели виртуального "черного ящика".

Литература

1. Варновский Н.П., Захаров В.А., Кузюрин Н.Н. Математические проблемы обфускации // Математика и безопасность информационных технологий. Материалы конференции в МГУ 28–29 октября 2004 г. Москва: 2005. С. 65–91.

2. Barak B., Goldreich O., Impagliazzo R., Rudich S., Sahai A., Vadhan S., Ke Yang. On the (im)possibility of obfuscating programs // Advances in Cryptology - CRYPTO'01, Lecture Notes in Computer Science, v. 2139, 2001, p. 1-18.

3. Dalla Preda M., Giacobazzi R. Semantic-based code obfuscation by abstract interpretation // International Colloquium on Automata, Language and Programming, Lecture Notes in Computer Science, v. 3580, 2005, p.1325-1336.

4. Козачок, А. В., Аналитическая модель защиты файлов документальных форматов от несанкционированного доступа / А. В. Козачок, М. В. Бочков, Р. Р. Фаткиева, Л. М. Туан. // Труды СПИИРАН (Санкт-Петербург), 2015.– Вып. 43. – С. 228–252.

5. Della Preda M., Giacobazzi G. Semantic-based code obfuscation by abstract interpretation // Journal of Computer Security, 2009, v. 17, N 6, p. 855-908.

6. Garg S., Gentry C., Halevi S., Raykova M., Sahai A., Waters B. Candidate indistinguishability obfuscation and functional encryption for all circuits // In FOCS, 2013. pp. 40–49.

7. Козачок А.В., Туан Л.М. Комплекс алгоритмов контролируемого разграничения доступа к данным, обеспечивающий защиту от несанкционированного доступа // Системы управления и информационные технологии. 2015. № 3(61). С. 58–61.

8. Ananth P., Gupta D., Ishai Y., Sahai A. Optimizing obfuscation: Avoiding barrington's theorem // In Proceeding of the 2014 ACM SIGSAC, 2014. pp. 646-658.

9. Pass R., Seth K., Telang S. Indistinguishability obfuscation from semantically-secure multilinear encodings // In Advances in Cryptology – CRYPTO, 2014. pp. 500–517.

10. Kilian J. Founding cryptography on oblivious transfer // In 20th Annual ACM Symposium on Theory of Computing 1988. pp. 20–31.

11. Garg S., Gentry C., Halevi S. Candidate multilinear maps from ideal lattices // In Advances in Cryptology – EUROCRYPT 2013. pp. 1–17.

12. Козачок А.В., Теоретическое обоснование стойкости неразличимой обфускации / А. В. Козачок, М. В. Бочков, Лай Минь Туан // Вопросы кибербезопасности. – 2016.– Вып. 1 (14).– С. 36–46.

А.Н. Крутов

ОПЫТ ВНЕДРЕНИЯ ЗАЩИЩЕННОЙ СИСТЕМЫ РЕПЛИКАЦИИ

(Самарский национальный исследовательский университет
имени академика С.П. Королёва)

В работе описывается опыт внедрения модуля «Бастион-2 – Репликация», являющегося частью аппаратно-программного комплекса «Бастион-2». Данный аппаратно-программный комплекс позволяет объединить системы безопасности различных производителей в области видеонаблюдения, контроля доступа и пожарной сигнализации в единую систему, удобным образом настраиваемую и надежно функционирующую. По своей сути аппаратно-программный комплекс «Бастион-2» является качественным развитием АПК «Бастион» с дополнительными возможностями. К ним относятся:

1. Переход от хранения данных в СУБД Firebird к СУБД Oracle. Для работы на относительно небольших объектах достаточно наличия бесплатной версии СУБД Oracle XE 11g и выше.
2. Существенные доработки ядра системы, позволяющие увеличить количество одновременно подключаемых драйверов.
3. Поддержка большего количества устройств, таких как биометрические считыватели, системы охраны периметра, модули аварийного освещения и т.п.

Кроме этого, АПК «Бастион-2» разрабатывался как масштабируемый продукт, способный к стабильной работе на объектах разного масштаба – от небольших офисов до крупных предприятий, с развитой филиальной сетью. Все вышесказанное послужило основанием для создания модуля «Бастион-2 – Репликация», основной функцией которого являлась бы передача электронных пропусков между объектами (филиалами) одного предприятия.

При анализе существующих решений по репликации данных в СУБД Oracle просматривались два решения, а именно технологий Oracle Streams и Golden Gate. Было отмечено, что использование технологии Oracle Streams требует весьма тщательной настройки сервера базы данных, требующей наличие на объекте администратора базы данных весьма высокой квалификации [1]. Сопровождение такой системы репликации неизбежно привело бы к необходимости привлечения существенных дополнительных ресурсов у службы технической поддержки. Кроме этого в работах [2,3] отмечается весьма ограниченные существующие средства по разрешению конфликтов репликации, что тре-