СТАТЬИ

УДК 004:003.26

КЛЮЧЕВЫЕ КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ НАД ОБЩЕЙ ПАМЯТЬЮ, И ТРАНСПОРТНЫЙ ПРОТОКОЛ. КОНТРОЛЬ ЦЕЛОСТНОСТИ ДИНАМИЧЕСКОЙ ОБЩЕЙ ПАМЯТИ МЕЖДУ ОТПРАВИТЕЛЕМ И ПОЛУЧАТЕЛЕМ В СИММЕТРИЧНОМ КРИПТОГРАФИЧЕСКОМ КАНАЛЕ

Александров А.В., Сорокин И.И.

Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых, Владимир, e-mail: alex_izi@mail.ru

Статья посвящена описанию ключевых и транспортных криптографических протоколов, разработанных для криптографических систем, использующих общую память Отправителя (А) и Получателя (В) в криптографическом симметричном канале секретной связи. Предложенные двусторонние протоколы охватывают решение следующих задач: создание симметричного ключа (ключевые протоколы), протокол обмена сообщениями (транспортные протоколы), протокол удаленного контроля целостности общей памяти на основе использования техники хеш-деревьев Дамгарда – Меркла с использованием транспортного протокола. Предложены два варианта протоколов генерации ключа – доверительный и проверяемый, отличающиеся степенью доверия между Отправителем (А) и Получателем (В). Разработанные протоколы достаточно устойчивы по отношению к MITM-атакам противника в канале связи. С использованием конструкции хеш-деревьев Дамгарда – Меркла существует принципиальная возможность уточнения несовпадающих значений общей памяти с помощью процедуры доказательства Меркла. Указано, что с введением динамической компоненты в общей памяти удается снять традиционно сложную проблему симметричной криптографии: передачи симметричного сеансового ключа от Отправителя (А) к Получателю (В), или создания симметричного ключа в результате проигрывания доверительного протокола, в определенном смысле аналога протокола Диффи – Хеллмана.

Ключевые слова: криптографический протокол, общая память, динамическая общая память, протокол Диффи – Хеллмана, хэш-функция, дерево Дамгарда – Меркла, доказательство Меркла

KEY CRYPTOGRAPHIC PROTOCOLS FOR SHARED MEMORY, AND TRANSPORT PROTOCOL. MONITOR THE INTEGRITY OF THE DYNAMIC SHARED MEMORY BETWEEN SENDER AND RECEIVER IN A IN A SYMMETRICAL CRYPTOGRAPHIC CHANNEL

Aleksandrov A.V., Sorokin I.I.

Vladimir State University named after Alexander and Nikolay Stoletovs, Vladimir, e-mail: alex_izi@mail.ru

The article is devoted to the description of key and transport cryptographic protocols developed for cryptographic systems using shared memory of the Sender (A) and Receiver (B) in the cryptographic symmetric channel of secret communication. The proposed bilateral protocols cover the following tasks: creation of a symmetric key (key protocols), message exchange protocol (transport protocols), protocol of remote control of the shared memory integrity based on the use of Damgard – Merkle hash tree technique using the transport protocol. Two variants of the key generation protocols are proposed – trustworthy and verifiable, differing in the degree of trust between the Sender (A) and Receiver (B). The developed protocols are sufficiently stable against enemy MITM-attacks in the communication channel. With the use of the design of Damgard – Merkle hash trees, there is a fundamental possibility of specifying different values of the shared memory with the help of Merkel's proof procedure. It is indicated that with the introduction of a dynamic component in the shared memory, it is possible to eliminate the traditionally complex problem of symmetric cryptography: the transfer of a symmetric session key from the Sender (A) to the Receiver (B), or the creation of a symmetric key as a result of playback of a confidential protocol, in a sense analogous to the Diffie-Hellman protocol.

Keywords: cryptographic protocol, shared memory, dynamic shared memory, Diffie-Hellman protocol, hash function, Damgard-Merkle tree, Merkle proof

В статье [1] представлены некоторые результаты по использованию общей памяти в криптографических протоколах, использующих симметричное шифрование, и, в частности, показано, что общую память отправителя и получателя можно использовать для противодействия MITM-атакам противника в канале связи. Метод применения общей памяти в криптографических протоколах сравнительно нов, поскольку выводит за пределы модели секретной связи К. Шеннона, и идеологию открытого распределения ключей. Однако использование ОП памяти не противоречит криптографической модели угроз Долева-Яо (1976 г.), в которой противник не имеет

доступа к устройствам, принадлежащим отправителю и получателю. В работе [1] также отмечено, что общий подход использования общей памяти и соответствующие протоколы – нуждаются в проработке и детализации.

Воспроизведенный в [1] протокол создания симметричного ключа на основе предварительного здесь мы называем доверительным. Такой протокол в потенциале своего использования может оказаться слаб по отношению к Получателю – В-лгуну – участнику протокола, меняющего корректные данные протокола и заинтересованного в корректном завершении его работы. Кроме того, В-лгун может входить в сговор с противником в канале связи, становясь более опасным, чем сам противник. Для закрытия подобной слабости здесь мы приводим проверяемый ключевой протокол, по своей симметрии похожий на протокол Диффи – Хеллмана, встроенного в TLS протоколы [2]. Проверяемый протокол, в отличие от протокола Диффи – Хеллмана, не использует трудность дискретного логарифма, а основан на невозможности противника в канале связи получить доступ к значениям общей памяти.

Деревья Дамгарда – Меркла, которые находят широкое применение в качестве базовой основы криптовалют Bitcoin и Etherium [3] и контроля целостности больших массивов данных, мы используем для построения протоколов контроля целостности и удаленной синхронизации общей памяти.

Цель исследования: Разработка криптографического протокола передачи и создания ключа с использованием общей памяти у отправителя и получателя, а также транспортного протокола. Такой протокол планируются как альтернатива протоколу Диффи – Хеллмана и его использование в гибридных криптографических системах. Предложения по удаленному контролю целостности общей памяти на устройствах отправителя и получателя.

Общая память и определения

Повторим основные определения из предыдущей статьи [1] и работы [4], и для этого обозначим здесь и далее $m > 1$ размер симметричного криптографического ключа k, $f_k^{\pm 1}(S)$ функции симметричного шифрования/расшифрования и пусть $H_k(S)$ – ключевая хэш-функция, сильно сопротивляющаяся поиску коллизий.

Определение 1. Назовем общей памятью Отправителя (А) и Получателя (В) – согласованное между Отправителем и Получателем и разнесенное на устройства Отправителя и Получателя числовое множество $D^A = \{d_1 \dots d_n\}$, $n > 1$, $n < m$. Верхний индекс X = {A,B} указывает, на каком устройстве располагается общая память.

Процедура создания общей памяти, выработки первого значения симметричного ключа и первого ключевого хэша не предполагает использования канала связи, и выполняется в момент создания множества D.

Определение 2. На основании общей памяти создадим предварительный ключ

$$E = (e_1 \dots e_n), e_i \in \{0,1\}, E \neq 0. \quad (1)$$

Сеансовый ключ генерируется по формуле

$$k_E = \sum_{i=1}^{n} e_i d_i \bmod 2^m, \quad (2)$$

где, напомним, m – размер сеансового ключа шифрования.

Утверждение 1. Пусть $D = \{d_1 \dots d_n\}$, $n > 1$, $n < m$ – общая память, созданная на устройствах отправителя и получателя ($D^A = D^B$) в модели секретной связи К. Шеннона, E – предварительный ключ, k_E – сеансовый ключ. Тогда противник С по значению предварительного ключа $E \neq 0$ не может построить симметричный ключ k_E. Это можно выразить в терминах условной энтропии К. Шеннона. $H(S|S_i)$: меры неопределенности значения S при условии обладания значением S_i

$$\begin{cases} H(k_E \mid \{E, D\}) = 0 \\ H(k_E \mid \{E\}) = H(k_E) - \delta(|E|) \end{cases}, \quad (3)$$

где $\delta(|E|) < H(k_E)$.

Доказательство использует технику (2-2) пороговых схем разделения секрета, для которых секретным значением является ключ k_E. Долями секрета являются соответственно значения $D = \{d_1 \dots d_n\}, n < m$, $E = \{e_1 \dots e_n\} \neq 0$.

В силу разбалансированности размеров долей секрета друг относительно друга, такая схема разделения секрета является неидеальной и несовершенной, но является пороговой. Из несовершенности вытекает, что противник С, перехватывая вектор Е получает некоторую информацию δ о значении ключа, но недостаточную, чтобы уточнить его значение. Техника несовершенных и почти совершенных схем разделения секрета исследована, в частности, в [5].

В терминах условной энтропии К. Шеннона, понимаемой относительно вероятностных распределений для значений k_E, E, D, первое равенство в (3.1) означает, что неопределенность значения ключа k_E по зна-

чениям теней E, D полностью снимается. Второе соотношение, важное для противника C в канале связи, определяет меру неопределенности симметричного ключа для противника C при перехвате предварительного ключа.

Точную оценку δ в терминах формул воспроизвести достаточно сложно, так как значения общей памяти заранее не определены. Но мы можем указать максимально возможное значение δ = 1. Для этого нами построены примеры. Пусть E – множество предварительных ключей, K – множество сеансовых ключей, которые можно получить с использованием формулы (2). Очевидно, мощность множества $|E| = 2^n - 1$, а множества $|K| = 2^m - 1$, здесь исключен случай нулевого вектора. В построенных нами примерах пересечением множеств предварительных и сеансовых ключей является пустое множество.

$$E \cap K = \varnothing. \quad (4)$$

Отметим, что согласно принципу Керхгоффса в криптографии, противник сможет знать и формулу (4). Это дает ему информацию о множестве сеансовых ключей, так что при брутфорсинге симметричных ключей он может отбросить случаи ключей из интервала $[1; 2^m]$ и проводить атаку на ключи на множестве $[2^n + 1; 2^m]$.

Далее, пусть распределение значений симметричных ключей из множества K – равномерно и равновероятно на $[2^n + 1; 2^m]$. В этом случае, хорошо известно, энтропийные оценки (3) принимают максимальные значения, в которых вместо условной энтропии участвует мера Хартли. Грубые оценки мер Хартли для множеств $[1; 2^m]$ и $[2^n + 1; 2^m]$ дают неравенство $\delta(|E|) \le 1$, где случай равенства достигается для случая $n = m - 1$. Это завершает доказательство утверждения 1.

Из утверждения 1 следует, что неопределенность значения симметричного ключа для противника C при перехвате предварительного ключа E снимается не более чем на один бит. В работе [6] описывается лгун в схемах разделения секрета.

Доверительный ключевой и транспортный протоколы

Доверительные протоколы создания симметричного ключа и передачи сообщения уже были предложены в статье [1]. Эти протоколы мы назвали доверительными, за счет того, что одна из сторон обмена сообщениями Отправитель A или Получатель B всецело доверяет противоположной стороне. Кто-либо из них может подменить данные при передаче сообщения. В протоколе (5) все проверки, и заключение об успешности завершения работы протокола происходят на стороне Получателя B, так что он в потенциале может быть лгуном.

Доверительный протокол создания симметричного ключа безопасный по отношению к противнику в канале связи:

1. $A \to B : E = \{e_1 \dots e_n\} \| H^A_{d_e}(E)$.
2. B: $H^B_{d_e}(E)$; если $H^b_{d_e}(E) \neq H^A_{d_e}(E)$, то стоп.
3. $A : k_{AB} = \sum e_i d_i \bmod 2^m ; H^A_{d_e}(k_{AB})$.
4. $B : k_{BA} = \sum e_i d_i \bmod 2^m ; H^B_{d_e}(k_{BA})$. (5)
5. $A \to B : H^A_{d_e}(k_{AB})$.
6. B: если $H^A_{d_e}(k_{AB}) = H^B_{d_e}(k_{BA})$, то $d_e \leftarrow k_{BA}$, иначе стоп.
7. $B \to A : H^B_{d_e}(Ok)$.
8. A: если $H^B_{d_e}(Ok) = H^A_{d_e}(Ok)$, то $d_e \leftarrow k_{AB}$.

Доверительный протокол обмена сообщениями:

1. $A \to B : f_k(S) \| H^A_k(S)$.
2. $B : f_k^{-1}\left(f_k(S) \| H^A_k(S)\right)$. (6)
3. B: если $H^B_k(S) = H^A_k(S)$, то Ok.

При корректном завершении работы протокола и равенстве значений хеш-функций на шаге (6.3), сообщение S считаем успешно переданным.

Протоколы (5), (6) эффективно противостоят MITM-атаке за счет использования ключевой хэш-функции H_{d_e} с ключом d_e – предыдущим сгенерированным сеансовым ключом. Активный противник, вмешиваясь в процесс передачи сообщений, может лишь портить пересылаемые данные, при этом Отправитель и Получатель на этапах (5.6) и (6.3) определят факт подмены сообщения.

Протокол (6) назовем транспортным протоколом, а фраза ТРАНСП(S) означает передачу сообщения S с помощью этого протокола.

Проверяемый ключевой протокол

Определение 3. Лгуном в протоколе называем законного участника протокола, заинтересованного в подмене данных в рамках действия протокола и корректном завершении работы протокола. Протоколы с лгунами естественным образом возникают в протоколах схем разделения секрета и впервые рассматривались в работе [7].

В предыдущих протоколах предполагается, что отправитель A выступает в роли «дилера» ключа, так как он генерирует предварительный ключ. В свою очередь получатель B осуществляет все сравнения хэш-значений в протоколах и ему же принадлежит роль определения корректности

завершения работы протоколов (5), (6). Роли A и B в протоколах неравноправны и основаны на значительном доверии A к B и соответственно B к A. Поэтому представленные выше протоколы мы можем назвать доверительными. В отличие от противника C, B-лгун имеет доступ к общей памяти D, что позволяет ему строить ключ $k_{BA} = k_{AB}$, а также знает ключ d_e и может получить $H_{d_e}(E) = H_{d_e}(E')$. Для противодействия B-лгуну необходимы усиления приведенных протоколов в виде двусторонних проверок передаваемой информации. Этим достигается определенная симметрия доверия A и B так, как это происходит в протоколе Диффи – Хеллмана:

1. $A \to B : a^x \bmod p$.
2. $B \to A : a^y \bmod p$. (7)
3. $A : k_{AB} = (a^y)^x \bmod p$.
4. $B : k_{BA} = (a^x)^y \bmod p$.

Как можно заметить, в (7) протоколе доверие между A и B критично, и в случае обмана одного из участников обмена сообщениями, общий ключ на шаге 4 корректно выработан не будет Протокол в определенном смысле симметричен по отношению доверительности участников протокола.

Обозначим $\oplus$ – оператор поразрядного сложения исключающего ИЛИ. Приведем проверяемый протокол, аналог протокола (7), устойчивый к атакам B-лгуна:

1. $A \to B : E = \{e_1 \ldots e_n\} \| H^A_{d_e}(E)$.
2. B: $H^B_{d_e}(E)$; если $H^b_{d_e}(E) \neq H^A_{d_e}(E)$, то стоп.
3. $A : k_{AB} = \sum e_i d_i \bmod 2^m ; H^A_{d_e}(k_{AB})$.
4. $A : k_{BA} = \sum e_i d_i \bmod 2^m ; H^B_{d_e}(k_{BA})$. (8)
5. $A \to B : H^A_{d_e}(k_{AB})$.
6. B: если $H^A_{d_e}(k_{AB}) = H^B_{d_e}(k_{BA})$, то $OK1$, иначе стоп.
7. $B \to A : H^B_{d_e}(OK1)$. (9)
8. A: Если $H^B_{d_e}(OK1) = H^A_{d_e}(OK1)$, то $K^A = (E^A \oplus E^B) d_i \bmod 2^m$, иначе стоп.
9. A: запрашивает $H^B_{d_e}(K^B)$.
10. $B : K^B = (E^B \oplus E^A) d_i \bmod 2^m$.
11. $B \to A : H^B_{d_e}(K^B)$.
12. A: Если $H^B_{d_e}(K^B) = H^A_{d_e}(K^A)$, то $d_e \leftarrow k_{BA}$, иначе стоп.
13. $A \to B : H^A_{d_e}(OK2)$.
14. A: Если $H^A_{d_e}(OK2) = H^B_{d_e}(OK2)$, то $d_e \leftarrow k_{BA}$, иначе стоп.

Утверждение 2. Пусть $H_{d_e}(S)$ сильно сопротивляется поиску коллизий, тогда в проверяемом ключевом протоколе ни A, ни B не могут быть лгунами без их обнаружения на другой стороне.

Динамическая общая память. Контроль целостности общей памяти

Обозначим дерево Дамгарда – Меркла DM^X, построенное над общей памятью соответственно Отправителя для $X = A$ и Получателя для $X = B$. Свойства этих деревьев хорошо известны и использованы в работе [1] для определения целостности общей памяти на устройствах A и B.

Ввиду изменения общей памяти за счет динамической компоненты становится особенно важно поддерживать целостность памяти. Контролировать общую память предлагается с использованием техники построения деревьев Дамгарда – Меркла. В таких деревьях корневой хэш Root_Hash представляет из себя результат попарной конкатенации всех вершин дерева. Очевидно, что, при совпадении корневых хэшей двух деревьев – совпадает и их общая память.

Для контроля целостности общей памяти Отправителя и Получателя предлагается использовать следующий протокол:

1. B: запрашивает Root_Hash(A).
2. A → B: ТРАНСП ($H^A_{d_e}$ (Root_Hash(A)).
3. B: сравнивает полученный Root_Hash(A) со своим Root_Hash(B). (10)

3.1. B: если Root_Hash(A) = Root_Hash(B), то OK(Shared_Memory(A) = Shared_Memory(B));

3.2. B: если Root_Hash(A) ≠ Root_Hash(B), то применяем ДОКАЗАТЕЛЬСТВО_МЕРКЛА (Shared_Memory(A) ≠ Shared_Memory(B)).

Протокол либо успешно завершает работу на этапе (3.1), когда корневые хэши Отправителя и Получателя равны, а значит и общая память в целом у них совпадает, либо на этапе (3.2) запускаем протокол доказательства Меркла, так как корневые хэши не совпадают, следовательно, общая память у Отправителя и Получателя различна.

ДОКАЗАТЕЛЬСТВО_МЕРКЛА:

1. B запрашивает хэши всех элементов общей памяти.
2. A → B: ТРАНСП ($H^A_{d_e}$ (Shared_Memory(A)[i])).
3. B: сравнивает полученный хэш $H^A_{d_e}$ (Shared_Memory(A)[i]) со своим $H^B_{d_e}$ (Shared_Memory(A)[i]). (11)

3.1. B: если $H^A_{d_e}$ (Shared_Memory(A)[i]) = $= H^B_{d_e}$ (Shared_Memory(A)[i]), то обращаемся к следующему элементу общей памяти;

3.2. B: если $H^A_{d_e}$ (Shared_Memory(A [i]) ≠ $\neq H^B_{d_e}$ (Shared_Memory(A)[i]) зануляем бит в предварительном ключе, соответствующий несовпадающему элементу памяти.

Проигрывание протоколов (10), (11) позволяет удаленно синхронизировать общую память на устройствах A и B. Сначала применяется протокол, который дает понять, нарушена ли целостность памяти (10). Если не нарушена – то достаточно только применения протокола (10), в противном случае – после протокола (10) применяем протокол (11) для игнорирования несовпадающих элементов общей памяти предварительным ключом, зануляя соответствующую компоненту предварительного ключа.

Динамическая общая память

В статье [1] указано, что противник C, постоянно вмешиваясь в проигрывание ключевых, может разрушать их работу, вызывая оператор стоп. Использование динамической компоненты в общей памяти, представленной в виде очереди, во многом может снять трудности выполнения протоколов. Динамическое изменение общей памяти зададим преобразованием

$$D = \{d_1 \dots d_k, d_{k+1} \dots d_n\} \to D = \\ = \{d_1 \dots d_k, S, d_{k+1} \dots d_{n-1}\}, \qquad (12)$$

где d_i – элемент общей памяти;
S – успешно переданное последнее сообщение.

Пример использования динамической общей памяти, позволяющий порождать новый сеансовый ключ, без проигрывания ключевых протоколов.

Пусть размер симметричного ключа шифрования m = 128. Количество элементов общей памяти n = 40.

Общая память D = {80, 65, 33, 86, 35, 22, 49, 95, 86, 33, 25, 80, 27, 68, 8, 44, 69, 28, 65, 21, 91, 2, 34, 74, 74, 35, 67, 1, 37, 72, 8, 93, 66, 43, 60, 66, 14, 15, 36, 60}.

В данном примере первые 30 элементов ($d_1 \dots d_{30}$) общей памяти постоянны и в статье [1] эту область памяти мы назвали статической компонентой, а последние 10 ({$d_{31} \dots d_{40}$} или {8, 93, 66, 43, 60, 66, 14, 15, 36, 60}) – динамической компонентой, имеющей структуру – очередь.

Предварительный ключ E = 101100010 11110010110110111000010101001l0. Получим сеансовый ключ на основании имеющихся данных по формуле (2): k = 1057.

Передадим с использованием транспортного протокола сообщение S = «101» и изменим общую память согласно (12). Динамическая компонента принимает следующий вид: {101, 8, 93, 66, 43, 60, 66, 14, 15, 36}.

Структура общей памяти изменится в соответствии с формулой (12).

Измененная общая память D' = {80, 65, 33, 86, 35, 22, 49, 95, 86, 33, 25, 80, 27, 68, 8, 44, 69, 28, 65, 21, 91, 2, 34, 74, 74, 35, 67, 1, 37, 72, 101, 8, 93, 66, 43, 60, 66, 14, 15, 36}.

Таким образом, у нас изменилась общая память, предварительный ключ остался прежним, следовательно, и мы имеем согласованное изменение значений сеансового ключа (2): k = 1138 на устройствах Отправителя и Получателя.

Заключение

Среди предложенных протоколов генерации симметричного ключа, конечно, желательно применять протокол (9) вместо (5), так как в нем происходит двусторонняя проверка между отправителем и получателем. Это позволяет обезопасить обмен сообщениями от лгуна – участника протокола, при двусторонней проверке он будет вычислен.

Использование динамической общей памяти (12) снимает одну из традиционных и трудных проблем симметричной криптографии – безопасную передачу и/или создание симметричного ключа по двустороннему криптографическому протоколу, заменяя проблему передачи/создания синхронным преобразованием общей памяти на двух устройствах. Однако здесь возникает следующая важная задача – постоянное поддержание целостности общей памяти на устройствах Отправителя и Получателя, для чего предлагается использовать протоколы (10), (11).

Благодаря протоколам (10), (11) мы можем удаленно контролировать общую память у Отправителя A и Получателя B. Протокол (10) позволяет определить, совпадает ли общая память на устройствах A и B, а протокол (11) – избавиться от несовпадающих элементов при создании предварительного ключа.

Работа выполнена при поддержке и финансировании Российского фонда фундаментальных исследований, грант № 18-01-00596А.

Список литературы

1. Сорокин И.И., Александров А.В. Протоколы встраивания общей памяти в симметричный канал секретной связи для противодействия MITM атакам // Современные наукоемкие технологии. 2019. № 9. С. 14–19.

2. Oppliger Rolf. Introduction. SSL and TLS: Theory and Practice. 2nd. Artech House 2016. P. 13.

3. Israa Alqassem, Davor Svetinovic. Towards Reference Architecture for Cryptocurrencies: Bitcoin Architectural Analysis. IEEE International Conference on Internet of Things (iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom). 2014. DOI: 10.1109/iThings.2014.78.

4. Александров А.В., Сорокин И.И. Симметричная рюкзачная криптографическая система с общей памятью, основанная на рекуррентных базисах в задаче об укладке рюкзаков // Известия высших учебных заведений. Приборостроение. 2019. Т. 62. № 4. С. 320–330.

5. Kurosawa K., Okada K., Sakano K., Ogata W., Tsujii S. Nonperfect secret sharing schemes and matroids, LNCS 765, Advances in Cryptology, Proceedings of Eurocrypt'93, Springer Verlag. 1993. P. 126–141.

6. Pasailă D., Alexa V., Iftene S. Cheating Detection and Cheater Identification in CRT-based Secret Sharing Schemes. International Journal of Computing 2010. Vol. 9. Iss. 2. P. 107–117.

7. Martin Tompa, Heather Woll. How To Share a Secret with Cheaters. Journal of Cryptology. 1988. P. 133–138.