3. *Девянин П. Н., Куликов Г. В., Хорошилов А. В.* Комплексное научно-обоснованное решение по разработке отечественной защищенной ОССН Astra Linux Special Edition // Методы и технические средства обеспечения безопасности информации: Материалы 23-й науч.-технич. конф. 30 июня–03 июля 2014 г. СПб.: Изд-во Политехн. ун-та, 2014. С. 29–33.
4. Операционные системы Astra Linux. `http://www.astra-linux.ru/`
5. Astra Linux. `https://ru.wikipedia.org/wiki/Astra\_Linux`
6. *Девянин П. Н., Кулямин В. В., Петренко А. К. и др.* О представлении МРОСЛ ДП-модели в формализованной нотации Event-B // Проблемы информационной безопасности. Компьютерные системы. 2014. № 3. С. 7–15.
7. *Devyanin P., Khoroshilov A., Kuliamin V., et al.* Formal verification of OS security model with Alloy and Event-B // LNCS. 2014. V. 8477. P. 309–313.

УДК 517.19 DOI 10.17223/2226308X/9/33

СХЕМА ОБЕСПЕЧЕНИЯ КОНФИДЕНЦИАЛЬНОСТИ В АЛГОРИТМЕ RAID-PIR

М. Р. Кащеев, Ю. В. Косолапов

Рассматривается задача обеспечения конфиденциальности информационной базы данных в схеме анонимного получения информации (private information retrieval) с удалённых серверов. Предполагается, что для хранения базы используются r серверов (r — нечётное), а для анонимного доступа к информации используется алгоритм RAID-PIR. Построен способ шифрования и распределения базы данных таким образом, чтобы, во-первых, по зашифрованным данным, хранящимся на каждом из серверов, нельзя было нарушить конфиденциальность базы данных, и, во-вторых, чтобы при чтении или перезаписи блока данных ни один из серверов не мог узнать, какой блок соответственно считывался или перезаписывался.

Ключевые слова: *анонимность данных, PIR, распределение данных.*

Под анонимностью в сетях передачи данных, как правило, понимается либо невозможность идентификации сервером пользователей, отправивших запрос (анонимность пользователя), либо невозможность идентификации сервером запрашиваемой пользователями информации (анонимность запроса) [1]. В настоящей работе рассматривается обеспечение второго варианта анонимности. Предполагается, что для хранения информационной базы данных используется несколько серверов. Пользователь заинтересован в получении некоторой части базы данных таким образом, чтобы серверы, участвующие в хранении, по отдельности не смогли идентифицировать, какая именно часть базы была запрошена пользователем. В подобных схемах серверы могут рассматриваться как недобросовестные наблюдатели, цель которых заключается в выяснении, в получении какой информации из базы данных заинтересован пользователь. Простейшим случаем обеспечения анонимности является схема с одним сервером, когда пользователь с сервера запрашивает всю базу полностью [2]. Обычно в системах обеспечения анонимности запроса предполагается, что серверу может быть известно (частично или полностью) информационное содержимое базы. В работе рассматривается ситуация, когда знание сервером базы нежелательно, при этом необходимо также защититься от получения сервером информации о расположении или доле запрашиваемой информации в базе. Таким образом, ставится задача построения схемы защиты

конфиденциальности информационной базы данных, позволяющей анонимно считывать данные из базы и перезаписывать их.

Пусть $\mathrm{DB} \in \mathbb{F}_2^n$ — информационная база данных, представленная в виде конкатенации b блоков длины k бит каждый: $\mathrm{DB} = (d_1, \ldots, d_b)$, $d_i \in \mathbb{F}_2^k$, а для хранения базы DB используются r серверов хранения $\mathrm{S}_1, \ldots, \mathrm{S}_r$, где r — нечётное число. Целью пользователей этой системы хранения является получение j-го ($j \in \{1, \ldots, b\}$) блока базы DB таким образом, чтобы каждый из серверов не узнал j. Для этого пользователь отправляет специальные вектор-запросы всем серверам, а серверы вычисляют побитовые суммы блоков, соответствующих вектор-запросу пользователя, и отправляют пользователю результаты суммирования. Вектор-запрос к серверу — это бинарный вектор, в котором единицы находятся на позициях с номерами блоков базы, которые сервер должен просуммировать в ответ. Отметим, что способ построения запросов и соответствующих ответов, при котором отдельный сервер может знать информационное содержимое базы, но не может по запросу понять, какой именно блок запрошен пользователем, предложен в [3]. Далее схему из [3] назовём RAID-PIR.

В настоящей работе на базе варианта RAID-PIR, когда каждый сервер отвечает несколькими блоками, строится решение задачи обеспечения конфиденциальности информации в базе относительно серверов. Для этого используется шифрование методом модульного гаммирования с $r-1$ случайными (псевдослучайными) ключами длины n. Предполагается, что для генерации ключевой последовательности (гаммы) используется генератор непредсказуемой последовательности чисел, который обозначим `rnd_gen`. Построены четыре протокола: распределения базы по серверам (**DistribDB**), чтения i-го блока из базы (**GetBlock**), перешифрования данных на серверах (**NewKey**) и перезаписи i-го блока в базе (**SetBlock**). Параметром протоколов является число p ($1 < p < r$) — число зашифрованных на разных ключах копий базы DB, хранимых на каждом сервере.

Алгоритм 1. DistribDB — распределение базы $\mathrm{DB} \in \mathbb{F}_2^n$ по серверам $\mathrm{S}_1, \ldots, \mathrm{S}_r$

1: Владелец базы строит набор $\Gamma = \{\Gamma^1, \ldots, \Gamma^r\}$, где для $j = 1, \ldots, r-1$ вектор $\Gamma^j = (\gamma_1^j, \ldots, \gamma_b^j)$, $\gamma_i^j \in \mathbb{F}_2^k$, генерируется с помощью генератора `rnd_gen`; $\Gamma^r := \bigoplus\limits_{i=1}^{r-1} \Gamma^i$.

2: По DB и Γ вычисляются векторы: $\mathrm{DB}_\Gamma^j = \mathrm{DB} \oplus \Gamma^j = (d_1^j, \ldots, d_b^j)$, $j = 1, \ldots, r$.

3: Серверу S_j передаются $\mathrm{DB}_\Gamma^{j \bmod (r+1) + \lfloor j/(r+1) \rfloor}, \ldots, \mathrm{DB}_\Gamma^{(j+p-1) \bmod (r+1) + \lfloor (j+p-1)/(r+1) \rfloor}$, $j = 1, \ldots, r$.

Алгоритм 2. GetBlock — получение i-го блока (блока d_i) из базы DB

1: Пользователь с помощью алгоритма RAID-PIR получает i-й блок d_i^j с каждого вектора DB_Γ^j, $j = 1, \ldots, r$.

2: Пользователь вычисляет: $\bigoplus\limits_{j=1}^{r} d_i^j = \bigoplus\limits_{j=1}^{r} (\gamma_i^j \oplus d_i) = \left(\bigoplus\limits_{j=1}^{r} \gamma_i^j \right) \oplus \left(\bigoplus\limits_{j=1}^{r} d_i \right)$. Так как $\bigoplus\limits_{j=1}^{r} \gamma_i^j = \mathbf{0} (\in \mathbb{F}_2^k)$ и r — нечётное число, то $\bigoplus\limits_{j=1}^{r} d_i^j = d_i$.

Алгоритм 3. NewKey — перешифрование данных на серверах с помощью набора $\widetilde{\Gamma} = \{\widetilde{\Gamma}^1, \ldots, \widetilde{\Gamma}^r\}$

1: Пользователь передаёт на сервер S_j, $j = 1, \ldots, r$, соответствующие этому серверу p векторов: $\widetilde{\Gamma}^{j \bmod (r+1)+\lfloor j/(r+1)\rfloor}, \ldots, \widetilde{\Gamma}^{(j+p-1) \bmod (r+1)+\lfloor (j+p-1)/(r+1)\rfloor}$.

2: Сервер S_j обновляет хранящиеся у него части: $\mathrm{DB}_\Gamma^l := \mathrm{DB}_\Gamma^l \oplus \widetilde{\Gamma}^l$, где $l \in \{j \bmod (r+1) + \lfloor j/(r+1)\rfloor, \ldots, (j+p-1) \bmod (r+1) + \lfloor (j+p-1)/(r+1)\rfloor\}$, $j = 1, \ldots, r$.

Алгоритм 4. SetBlock — перезапись i-го блока в базе DB новым значением $\widetilde{d}_i$

1: Пользователь получает i-й блок базы ($d_i = \mathbf{GetBlock}(i)$), генерирует новый набор $\widetilde{\Gamma} = \{\widetilde{\Gamma}^1, \ldots, \widetilde{\Gamma}^r\}$ ($\widetilde{\Gamma} \leftarrow$ `rnd_gen`) и для каждого $\widetilde{\Gamma}^j = (\widetilde{\gamma}_1^j, \ldots, \widetilde{\gamma}_b^j)$ из $\widetilde{\Gamma}$ переопределяет $\widetilde{\gamma}_i^j$: $\widetilde{\gamma}_i^j := \widetilde{\gamma}_i^j \oplus (\widetilde{d}_i \oplus d_i)$.

2: Пользователь выполняет протокол перешифрования $\mathbf{NewKey}(\widetilde{\Gamma})$.

Показано, что

1) протоколы **GetBlock** и **SetBlock** обеспечивают анонимность соответственно запрашиваемых и записываемых данных;
2) протоколы **DistribDB**, **NewKey** и **SetBlock** обеспечивают конфиденциальность хранимых на серверах данных;
3) любая коалиция мощности $t < \lceil r/p \rceil$ не может нарушить конфиденциальность базы данных, а любая коалиция мощности $t \geqslant r - p + 1$ однозначно дешифрует базу данных.

ЛИТЕРАТУРА

1. *Pfitzmann A. and Hansen M.* A terminology for talking about privacy by data minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management. `https://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.18.pdf`. Дата обращения 30.03.2016.
2. *Chor B., Goldreich O., Kushilevitz E., and Sudan M.* Private information retrieval // J. ACM. 1998. V. 45(6). P. 965–981.
3. *Demmler D., Herzberg A., and Schneider T.* RAID-PIR: Practical Multi-Server PIR // Proc. 6th edition of the ACM Workshop on Cloud Computing Security. N. Y., USA: 2014. P. 45–56.

УДК 004.94 DOI 10.17223/2226308X/9/34

МЕТОД ЗАПУТЫВАНИЯ ПРОГРАММНОЙ РЕАЛИЗАЦИИ СХЕМЫ HMAC ДЛЯ НЕДОВЕРЕННОЙ СРЕДЫ

Д. Н. Колегов, О. В. Брославский, Н. Е. Олексов

Предлагается метод обфускации схемы аутентификации сообщений HMAC для реализации в недоверенных средах.

Ключевые слова: *white-box cryptography, коды аутентификации сообщений, HMAC, обфускация, защита приложений.*

При разработке защищённых веб-приложений часто необходимо реализовывать алгоритмы выработки кодов аутентификации сообщений (MAC) на языке JavaScript