ISSN 2079-3316 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ №1(24), 2015, с. 135–145

УДК 519.682.3

А. А. Кузнецов

Исследование криптостойкости протокола аутентификации BotikKey к компрометации уязвимостей алгоритма хеширования MD5

Аннотация. В работе приведен анализ уязвимостей сетевого протокола BotikKey, который используется в системе телекоммуникаций «Ботик» г. Переславля-Залесского для аутентификации абонентских подключений. Протокол разработан в рамках подхода Ботик-технологий, согласно которому все программно-аппаратное обеспечение сети «Ботик» является либо свободно-распространяемым, либо разработано собственными усилиями компании-провайдера. Дано описание назначения протокола и деталей реализации, и перечислены уязвимости, связанные с использованием алгоритма хеширования MD5. Приведены возможные способы компрометации протокола BotikKey, в том числе APOP-атака, целью которой является подбор пароля доступа. Даны рекомендации провайдеру услуг связи системы телекоммуникаций «Ботик» по отказу от системы BotikKey, либо переходу на более актуальные средства аутентификации абонентов.

Ключевые слова и фразы: протокол BotikKey, безопасная аутентификация, криптография, алгоритм MD5, APOP-атака.

Введение

В телекоммуникационной сети «Ботик» г. Переславля-Залесского клиент-серверный протокол BotikKey применяется с 2003 года для аутентификации абонентов. С 2007 года использование клиентской программы BotikKey из набора программ «BotikTools» [1, 2] стало обязательным для постоянных подключений абонентов СТ «Ботик».

Данная работа посвящена описанию протокола и встроенных в него методов защиты от компрометации. В данном контексте под компрометацией понимается ситуация, при которой злоумышленнику удалось выдать себя за добросовестного абонента и аутентифицировать свой компьютер на BotikKey-сервере.

1. Ключевые слова и термины

Запрос (*challenge*)**:** key-сервер присылает программе BotikKey запрос, на который от имени абонента нужно правильно ответить, используя пароль (ключ) доступа данного абонента к сети.

Ответ (*response*)**:** по полученному запросу и по паролю (ключу) доступа программа BotikKey строит ответ и передает его Key-серверу.

Используются принципы, основанные на том, что если известен правильный пароль можно построить правильный ответ и тем самым подтвердить право абонента на доступ к сети. Если на свой запрос Key-сервер получает правильный ответ (то есть, ответ соответствует паролю абонента), то Key-сервер признает, что с ним общается именно «тот самый абонент». В противном случае Key-сервер отвергает попытку выдать себя за абонента и перекрывает доступ к сети.

Пропуск в сеть: в случае признания Key-сервером ответа, абонент получает право пользования сетью (при наличии неизрасходованной предоплаты на лицевом счету абонента).

Пароль и ключ доступа: пароль обычно подбирается таким, чтобы он был достаточно сложным, но легко запоминался бы пользователем.

Для хранения на диске в файле хранится не пароль, а построенная по паролю некоторая строка символов — ключ. Программе BotikKey достаточно либо задать пароль, либо предоставить файл с ключом доступа — в обоих случаях она с одинаковым успехом будет строить ответы на запросы Key-сервера.

При помощи программы BotikKey можно запросить у Key-сервера различный **режим** (*регион*) **доступа** для своего компьютера**:**

WORLD: разрешение для данного компьютера как на внутренние (внутри СТ «Ботик»), так и на внешние обмены;

LOCAL: разрешение для данного компьютера только внутренних (внутри СТ «Ботик») обменов, запрет на внешние обмены;

NONE: запрет для данного компьютера на все обмены: как внутренние (внутри СТ «Ботик»), так и внешние.

Абонент может иметь несколько паролей доступа; у каждого пароля определяется свой предельный режим доступа: WORLD, LOCAL или NONE.

WORLD-ключ: имея пароль с предельным режимом WORLD, можно успешно запросить у Key-сервера любой доступ (WORLD, LOCAL или NONE).

LOCAL-ключ: имея пароль с предельным режимом LOCAL, можно успешно запросить у Key-сервера только доступ LOCAL или NONE, а доступ WORLD получить невозможно: если при помощи LOCAL-ключа будет запрошен доступ WORLD, то Key-сервер предоставит доступ LOCAL.

NONE-ключ: имея пароль с предельным режимом NONE, можно успешно запросить у Key-сервера только доступ NONE, а доступ WORLD или LOCAL получить невозможно: если при помощи LOCAL-ключа будет запрошен доступ WORLD или LOCAL, то Key-сервер предоставит только доступ NONE.

2. Сетевой протокол BotikKey

В СТ «Ботик» клиент-серверный текстовый протокол BotikKey применяется при аутентификации подключений абонентов. В описании протокола используются следующие обозначения:

***random*:** случайная строка из алфавита — .0123456789

***region*:** запрашиваемый регион доступа — одна из строк: WORLD, LOCAL, NONE;

***challenge*:** запрос, как правило представляющий собой MD5-сумму от случайной строки;

***accessKey*:** ключ доступа, представляющий собой хеш-сумму от пароля BotikKey (результат применения к паролю функции хеширования MD5, запись в 16-ричном формате);

***salt*:** конкатенация строк *challenge*·*random*·*region*.

Аутентификация клиента на сервере происходит каждую минуту по следующей схеме:

(1) клиент устанавливает незащищенное socket-соединение с Key-сервером;

(2) сервер посылает клиенту строку-приветствие и затем строку

BotikKey server:

```
1 CHALLENGE challenge
```

запроса;

(3) клиент посылает в формате

BotikKey client:

```
1 RESPONSE  secret random region
```

ответ, в котором $secret$=MD5($salt{\cdot}accessKey$) это результат применения хеш-алгоритма MD5 к конкатенации строк;

(4) сервер проверяет эту информацию: первый аргумент у RESPONSE должен совпадать с хеш-суммой, вычисленной на Key-сервере от этой же конкатенации строк. Если совпадение произошло, то сервер отвечает

BotikKey server:

```
1 GOOD region
```

и клиент аутентифицирован. В противном случае доступ в сеть перекрывается и клиенту отправляется строка SORRY, иногда с указанием конкретной причины.

Key-сервер как правило установлен на шлюзе (IP-gateway). Успех аутентификации клиента на сервере означает, что ему предоставляется доступ в сеть Интернет. Ошибка аутентификации означает перекрытие каналов связи для данного подключения. Регулирование доступа производится на Linux-шлюзе средствами iptables.

3. Алгоритм хеширования MD5

Хеш-функции представляют собой односторонние функции, преобразующие входную строку произвольной длины в строку фиксированной длины (хеш-сумму). Хеш-сумму можно рассматривать как подпись исходного сообщения, что, например, позволяет после передачи информации по каналам связи проверить подлинность переданной информации. Хеш-функции спроектированы так, чтобы было просто вычислить хеш-сумму сообщения, но было бы тяжело восстановить исходное сообщение. Они имеют широкий диапазон применений: аутентификация, проверка целостности сообщений, цифровые сертификаты и цифровые подписи.

Безопасность применений хеш-функции зависит от ее криптостойкости. Чтобы хеш-функцию можно было считать криптостойкой, необходимо соблюдение нескольких условий:

(1) устойчивость к атакам поиска первого прообраза: по заданной хеш-сумме h должно быть тяжело подобрать такое сообщение m, чтобы $H(m) = h$;

(2) устойчивость к атакам поиска второго прообраза: по заданному сообщению m_1 должно быть тяжело подобрать другое сообщение m_2 так, чтобы $H(m_1) = H(m_2)$;
(3) устойчивость к коллизиям: должно быть тяжело подобрать разные сообщения m_1 и m_2 такие, чтобы $H(m_1) = H(m_2)$.

Поскольку область определения хеш-функции намного шире чем область значений, то в соответствии с принципом Дирихле должно существовать множество коллизий. Атака методом «грубой силы» может обнаружить 1-й или 2-й прообраз для хеш-функции с n-разрядной хеш-суммой примерно за 2^n операций хеширования. На самом деле, из-за «парадокса дней рождения» коллизия будет найдена примерно за $2^{n/2}$ операций хеширования. Если можно провести атаку на хеш-функцию, которая требует меньше вычислений, чем полный перебор, то хеш-функцию можно считать скомпрометированной.

Хеш-функция MD5 [3] основана на структуре Меркла–Дамгарда. Она использует функцию сжатия, которая на входе принимает k-разрядный блок исходного текста и n-разрядное промежуточное хеш-значение (IHV) и на выходе порождает новое промежуточное хеш-значение. Исходное сообщение перед хешированием дополняется до длины кратной k, и разбивается на блоки по k бит. Затем функция сжатия применяется последовательно к каждому блоку, начиная с некоторого фиксированного начального состояния IHV_0. В алгоритме MD5 применяются блоки по 512 бит и 128-разрядные хеш-значения. Хеш-функция работает с 32-разрядными целыми числами, с использованием операции сложения по модулю 2^{32} и побитовыми операциями.

4. Уязвимости алгоритма хеширования MD5

В 2004 году исследователями было показано [4, 5] что алгоритм хеширования MD5 не обладает достаточной устойчивостью к коллизиям. Сообщения (последовательности байт) M и M' образуют коллизию под MD5, если хеш-сумма $\mathrm{MD5}(M)$ побитово совпадает с $\mathrm{MD5}(M')$. Исследователями были описаны и реализованы на практике дифференциальные методы генерации двух байтовых последовательностей (с возможностью указания любого начального вектора), у которых совпадают хеш-суммы.

Задание начального вектора при генерации коллизии позволяет злоумышленнику вставить каждое из сообщений (M, M') внутрь двух

различных файлов в определенных позициях внутри них. Это приводит к созданию, например, двух Postscript-документов, имеющих одну хеш-сумму, но разный отображаемый текст. Также, продемонстрированы методы генерации двух исполняемых файлов, имеющих одну хеш-сумму, но разное поведение. По причине неустойчивости к коллизиям различные исследовательские организации по всему миру рекомендуют отказаться от использования алгоритма хеширования MD5.

Методы обеспечения безопасной передачи данных в протоколе BotikKey почти аналогичны тем методам, которые применяются в протоколе APOP [6], который некоторые почтовые клиенты до сих пор используют для усиленной авторизации на почтовых POP-серверах. Принцип действия APOP-аутентификации напоминает представленную выше схему «запрос-ответ» и тоже использует MD5:

(1) Почтовый клиент устанавливает соединение к почтовому серверу.
(2) Сервер отправляет в формате

BotikKey server:
```
PID.clock@hostname
```

запрос в адрес клиента, где:
PID — идентификатор процесса,
clock — метка времени,
hostname — доменное имя сервера.
(3) Клиент посылает в формате

BotikKey client:
```
APOP username hashsum
```

ответ серверу, где:
username — логин пользователя на почтовом сервере,
hashsum — MD5-сумма от конкатенации строки PID.*clock*@*hostname* и пароля пользователя.
(4) После получения ответа клиента сервер со своей стороны вычисляет хеш-сумму MD5(PID.*clock*@*hostname*·*passwd*), где *passwd* представляет собой известный на сервере пароль данного пользователя, и производит сравнение хеш-сумм.
(5) Если хеши совпадают, то клиент успешно проходит аутентификацию и получает доступ к своей почте.

Достоинство этого подхода заключаются в том, что злоумышленник, перехвативший сообщения клиента и сервера, не сможет узнать пароль пользователя и тем самым получить доступ к его почте.

В настоящее время протокол APOP считается скомпрометированным, так как его применение позволит злоумышленнику (который выдает себя за почтовый сервер) за разумное время восстановить до 43 символов (байт) пароля [7]. Опишем стратегию, применяемую для выполнения атаки на протокол APOP. Она основана на том, что выбирается пара сообщений (m, m'), которая образует коллизию под $MD5$. Тогда для любой строки x выполняется равенство

$$\mathrm{MD5}(m \cdot x) = \mathrm{MD5}(m' \cdot x).$$

Принцип атаки состоит в том, что выбирается начальный символ пользовательского пароля, генерируется коллизия (m, m'), где каждое сообщение содержит этот символ в качестве суффикса. Затем подобранный символ отрезается от сообщений (m, m') и результат посылается как «запрос» ничего не подозревающему клиенту. Клиент вычисляет дайджесты от двух сообщений·пароль и посылает их серверу. Если дайджесты совпадают, то первый символ угадан, можно подбирать второй и последующие символы пароля. Приведем более формальное описание схемы атаки:

(1) Сгенерировать пару сообщений (m, m'), которые образуют MD5-коллизию, такую, что последние несколько байт каждого сообщения представляют собой найденные начальные символы пароля, в которых первые знаки уже подобраны правильно (являются префиксом пароля), а последний знак s является кандидатом на вхождение в найденную последовательность. Пусть C — это сообщение m без угаданных символов, и пусть C' — это сообщение m' без угаданных символов, тогда послать пользователю строку C в качестве запроса.
(2) Получить ответную хеш-сумму R, затем отправить C' в качестве другого запроса пользователю.
(3) Получить ответную хеш-сумму R' и сравнить между собой R и R'. Если произошло совпадение, то символ s угадан правильно, можно добавить его в конец угаданной последовательности. Если совпадения не произошло, то перейти к шагу 1 следующей попытки.

Отличия протокола APOP от BotikKey состоят в том, что у APOP *соль* имеет более простой формат — одна строка *challenge*, в то время как в BotikKey *соль* включает в себя также случайную строку *random* и регион доступа (*соль* = *challenge*·*random*·*region*).

С применением механизма «туннелей» [8] поиск двухблочных коллизий на современном персональном компьютере занимает менее одной минуты. Были также разработаны методы поиска 512-разрядных (одноблочных) коллизий MD5 [9], и методы подбора коллизий с заданными префиксами [10], которые на практике занимают гораздо большее время.

Схему APOP-атаки теоретически можно применить для компрометации протокола BotikKey. Сделав некоторые допущения, при соблюдении определенных условий можно произвести APOP-атаку на абонентское подключение, использующее протокол BotikKey, с целью подбора ключа доступа. Если не принимать во внимание защитные механизмы, встроенные в сетевую аппаратуру СТ «Ботик» для абонентских подключений, APOP-атака будет успешно завершена за разумное время.

Поскольку автор данного исследования специализируется на разработке параллельных приложений для супер-ЭВМ, у него возник закономерный вопрос: можно ли распараллелить программу подбора одноблочных MD5-коллизий? Ответ: можно; эта работа была выполнена. Для проведения вычислительных экспериментов автор разработал параллельную MPI-версию программы Марка Стивенса, которая выполняет подбор одноблочной MD5-коллизии.

Заключение

По результатам проведенного исследования можно сделать следующие выводы:

(1) Вероятность компрометации протокола BotikKey тем выше, чем слабее BotikKey-пароль у абонента.
(2) Уязвимость алгоритма хеширования MD5 к коллизиям не влечет за собой непосредственную угрозу пользователям протокола BotikKey. Но в будущем могут быть открыты новые более эффективные криптографические методы поиска MD5-коллизий, что сделает использующие его системы и протоколы еще более уязвимыми.
(3) От протокола BotikKey следует либо полностью отказаться, либо перейти на использование более криптостойких хеш-функций.

Протокол BotikKey на момент публикации этой работы не представляется уязвимым к атакам с использованием коллизий MD5. Но с

ростом вычислительных мощностей современных супер-ЭВМ положение дел может измениться. Поэтому автор рекомендует подобрать более современные методы криптозащиты для протокола BotikKey. Эти методы могут основываться на использовании SHA-3 [**11**] или иных криптоустойчивых алгоритмов.

Благодарности. *Работы, положенные в основу данной статьи, были выполнены в рамках НИР «Методы и программные средства разработки параллельных приложений и обеспечения функционирования вычислительных комплексов и сетей нового поколения». Автор выражает благодарности разработчикам системы и протокола BotikKey (С. М. Абрамов, Ю. В. Шевчук) за полезные комментарии по содержанию статьи.*

Список литературы

[1] *Пакет программ BotikTools*, URL http://www.botik.ru/~botik/tools/index.ru.html ↑ 135.

[2] С. М. Абрамов, А. А. Кузнецов, «BotikTools — пакет программ для Абонентов научно-образовательной сети г. Переславля-Залесского», *Международная конференция «Программные системы: теория и приложения»*. Т. 1 (Переславль-Залесский, октябрь 2006), Наука. Физматлит, М., 2006, с. 135–154 ↑ 135.

[3] R. L. Rivest.. *The MD5 message-digest algorithm*, Request for Comments (RFC 1320), Internet Activities Board, Internet Privacy Task Force, 1992 ↑ 139.

[4] X. Wang, D. Feng, X. Lai, H. Yu. *Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD*, Cryptology ePrint Archive, Report 2004/199, 2004, URL http://eprint.iacr.org/2004/199 ↑ 139.

[5] X. Wang, H. Yu, “How to Break MD5 and Other Hash Functions”, Eurocrypt’05, LNCS, vol. **3494**, Springer-Verlag, 2005, pp. 19–35 ↑ 139.

[6] J. Myers, M. Rose. *RFC 1939. Post Office Protocol*, Version 3, May 1996, URL http://tools.ietf.org ↑ 140.

[7] F. Liu, Y. Liu, T. Xie, D. Feng, Y. Feng. “Fast password recovery attack: application to APOP”, *Journal of Intelligent Manufacturing*, **25**:2 (2014), pp. 251–261 ↑ 141.

[8] V. Klima. *Tunnels in Hash Functions: MD5 Collisions Within a Minute*, Cryptology ePrint Archive, Report 2006/105, 2006, URL https://eprint.iacr.org/2006/105 ↑ 142.

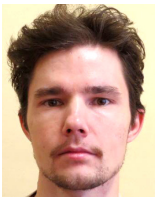
[9] M. Stevens. *Single-block collision attack on MD5*, Cryptology ePrint Archive, Report 2012/040, http://eprint.iacr.org/2012/040 ↑ 142.

[10] M. Stevens, A. K. Lenstra, B. de Weger. “Chosen-prefix collisions for MD5 and applications”, *Int. J. Applied Cryptography*, **2**:4 (2012), pp. 322–359 ↑ 142.

[11] *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*, URL http://csrc.nist.gov/publications/drafts/fips202/fips_202_draft.pdf ↑ 143.

Рекомендовал к публикации *д.ф.-м.н. С. В. Знаменский*

Об авторе:



Антон Александрович Кузнецов

Научный сотрудник ИЦМС ИПС им. А.К. Айламазяна РАН. Разработчик всех программ пакета BotikTools. Один из разработчиков системы OpenTS. Область научных интересов: системы параллельного программирования, распределенные вычисления в гетерогенных средах, геоинформационные системы.

e-mail: tonic@pereslavl.ru

Пример ссылки на эту публикацию:

А. А. Кузнецов. «Исследование криптостойкости протокола аутентификации BotikKey к компрометации уязвимостей алгоритма хеширования MD5», *Программные системы: теория и приложения*, 2015, **6**:1(24), с. 135–145.

URL http://psta.psiras.ru/read/psta2015_1_135-145.pdf

Anton Kuznetsov. *On the cryptographic security of the "BotikKey" authentication protocol against attacks on MD5 hash function.*

Abstract. In this paper vulnerabilities of the BotikKey network protocol are described. It is being used in the "Botik" telecommunication system of Pereslavl-Zalesskiy for secure subscribers' authentication. Protocol was developed as part of Botik-technologies initiative, according to which all software and hardware is based on open source, or on the inhouse developments. We outline the purpose and implementation details of the protocol and its pros and cons. It is pointed out that majority of the protocol's vulnerabilities arise from the weaknesses of MD5 cryptographic hash function being used. With a number of assumptions, the BotikKey protocol can be compromised by committing an APOP-attack on a subscriber. It is noted that "Botik" network service provider should use contemporary cryptographic methods for subscribers' authentication or avoid using the BotikKey system at all. (*In Russian*).

Key Words and Phrases: BotikKey protocol, secure authentication, cryptography, MD5 hash function, APOP attack.

Sample citation of this publication

Anton Kuznetsov. "On the cryptographic security of the "BotikKey" authentication protocol against attacks on MD5 hash function", *Program systems: theory and applications*, 2015, **6**:1(24), pp. 135–145. (*In Russian.*)

URL http://psta.psiras.ru/read/psta2015_1_135-145.pdf