УДК 621.391

ЗАЩИТА УДАЛЕННОГО ДОСТУПА К СЕРВЕРУ ОТ АТАКИ МЕТОДОМ «ГРУБОЙ СИЛЫ»

Р.М. ГОРБУЛЬ, Д.Д. ТРОФИМЕНКО, Ф. АЛИХАНОВ

Белорусский государственный университет информатики и радиоэлектроники
П. Бровки, 6, Минск, 220013, Беларусь

Поступила в редакцию 23 октября 2015

Рассмотрена атака методом «грубой силы». Произведена оценка факторов риска атаки. Рассмотрены методики защиты удаленного доступа. Показана эффективность методик защиты от атаки методом «грубой силы».

Ключевые слова: удаленный доступ, метод «грубой силы», защита от атаки методом «грубой силы».

Введение

Задача обеспечения безопасного удаленного доступа к серверу крайне важна. Одной из распространенных атак на службу ssh является атака методом «грубой силы» (метод перебора). Цель данной атаки – удаленно получить доступ к аккаунтам пользователей путем многократных попыток угадать пароль пользователя или группы пользователей. Если веб-сервер не имеет никаких защитных мер против этого типа атак, то злоумышленнику довольно просто взломать систему, основанную на парольной аутентификации, осуществив сотню попыток ввода пароля с помощью автоматизированных программ.

Целью данной работы является исследование методик защиты удаленного доступа по ssh от атаки методом «грубой силы» и их применения.

Атака методом «грубой силы»

Хотя метод «грубой силы» в большинстве прикладных задач (особенно не связанных со взломом) на практике не применяется, но есть ряд исключений. В частности, когда данный метод все же оказывается оптимальным, либо представляет собой начальный этап в разработке алгоритма, его использование оправдано. Примером оптимальности «грубой силой» является алгоритм оценки времени вычисления цепочечных произведений матриц, который не удается ускорить по сравнению с алгоритмом, основанным на методе «грубой силы». Этот алгоритм используется для решения классической задачи динамического программирования – определения приоритетов вычислений матричных произведений следующего вида: $A_1A_2A_3\ldots A_n$.

Исходная задача заключается в вычислении данной цепочки (матричного произведения) за наименьшее время. Можно реализовать тривиальный последовательный алгоритм, вычисляющий искомое произведение. Поскольку матричное произведение является ассоциативной операцией, можно вычислить цепочечное произведение, произвольно выбирая пару элементов цепочки $\left(A_iA_{i+1}\right), i=1\ldots n-1$ и заменяя ее результирующей матрицей $A_i^1 : A_i^1 = \left(A_iA_{i+1}\right)$. Если повторять описанную процедуру $n-1$ раз, то оставшаяся результирующая матрица A_k^{n-1} и будет ответом: $A_k^{n-1} = \left(A_k^{n-2} \cdot A_{k+1}^{n-2}\right) = \ldots = A_1A_2A_3\ldots A_n,\ k=1\ldots n-1.$ Эта формула может быть проиллюстрирована следующим образом. Рассмотрим матричную цепочку: $\langle A_1A_2A_3A_4\rangle$.

Существуют следующие 5 способов вычисления соответствующего этой цепочке произведения $A_1A_2A_3A_4$:

$\left(A_1\left(A_2\left(A_3A_4\right)\right)\right)$

$\left(A_1\left(\left(A_2A_3\right)A_4\right)\right)$

$\left(\left(A_1A_2\right)\left(A_3A_4\right)\right)$

$\left(\left(A_1\left(A_2A_3\right)\right)A_4\right)$

$\left(\left(\left(A_1A_2\right)A_3\right)A_4\right)$

Выбрав правильный порядок вычислений, можно добиться значительного ускорения вычислений. Чтобы убедиться в этом, рассмотрим простой пример цепочки из 3-х матриц. Положим, что их размеры равны соответственно 10×100, 100×5, 5×50. Стандартный алгоритм перемножения двух матриц размерами $p \times q$, $q \times r$ требует время вычисления, пропорциональное числу pqr (число вычисляемых скалярных произведений). Следовательно, вычисляя цепочку в порядке $\left(\left(A_1A_2\right)A_3\right)$, получаем $10 \cdot 100 \cdot 5 = 5000$ скалярных произведений для вычисления $\left(A_1A_2\right)$, плюс дополнительно $10 \cdot 5 \cdot 50 = 2500$ скалярных произведений, чтобы вычислить второе матричное произведение. Общее число скалярных произведений: 7500. При ином выборе порядка вычислений получаем $100 \cdot 5 \cdot 50 = 25000$ плюс $10 \cdot 100 \cdot 50 = 50000$ скалярных произведений, то есть 75000 скалярных произведений.

Таким образом, решение данной задачи может существенно сократить временные затраты на вычисление матричной цепочки. Это решение может быть получено полным перебором: необходимо рассмотреть все возможные последовательности вычислений и выбрать из них ту, которая при вычислении цепочки занимает наименьшее число скалярных произведений. Однако надо учитывать, что этот алгоритм сам по себе требует экспоненциального времени вычисления, так что для длинных матричных цепочек выигрыш от вычисления цепочки самым эффективным образом (оптимальная стратегия) может быть полностью потерян временем нахождения этой стратегии.

В криптографии на полном переборе основывается криптографическая атака методом «грубой силы». Ее особенностью является возможность применения против любого практически используемого шифра. Однако такая возможность существует лишь теоретически, зачастую требуя нереалистичные временные и ресурсные затраты. Наиболее оправдано использование атаки методом «грубой силы» в тех случаях, когда не удается найти слабых мест в системе шифрования, подвергаемой атаке (либо в рассматриваемой системе шифрования слабых мест не существует). При обнаружении таких недостатков разрабатываются методики криптоанализа, основанные на их особенностях, что способствует упрощению взлома.

Устойчивость к атаке методом «грубой силы» определяет используемый в криптосистеме ключ шифрования. Так, с увеличением длины ключа сложность взлома этим методом возрастает экспоненциально. В простейшем случае шифр длиной в N битов взламывается, в наихудшем случае, за время, пропорциональное $2N$. Среднее время взлома в этом случае в два раза меньше и составляет $2N-1$. Существуют способы повышения устойчивости шифра к атаке методом «грубой силы», например запутывание (обфускация) шифруемых данных, что делает нетривиальным отличие зашифрованных данных от незашифрованных.

Методика защиты от атаки методом «грубой силы»

Для обеспечения безопасного удаленного подключения к серверу необходимо произвести корректные настройки демона ssh, а также встроенного брандмауэра iptables. Также в качестве дополнительной меры безопасности можно использовать программное обеспечение psad и методику port knocking.

Несмотря на то, что протокол ssh использует шифрование, для безопасного использования требуется корректная настройка ssh-сервера. Одним из самых уязвимых мест доступа с

использованием ssh является доступ по пользовательскому паролю. Такой доступ подвержен атакам методикой «грубой силы». Любой веб-сервер или сервер с прямым доступом в интернет ежедневно сканируется на наличие открытых портов с дальнейшим подбором учетных данных с целью получения доступа. Подбор осуществляется, как правило, на основе заранее созданных словарей с наиболее используемыми учетными данными. Для предотвращения несанкционированного доступа настоятельно рекомендуется отказаться от использования парольного доступа и использовать только аутентификацию по публичным ключам. Также не менее важным правилом является запрет удаленного доступа к учетной записи суперпользователя – «root». В качестве дополнительной меры безопасности можно указать список разрешенных пользователей. Также существует ложное правило, которое потенциально является не мерой безопасности, а лишь потенциальной уязвимостью. Правило смены порта ssh – 22, на случайный непривилегированный порт. Данное правило является потенциальной уязвимостью, так как в ОС на основе ядра Linux привилегированные порты могут быть открыты только от имени суперпользователя, а непривилегированные могут использоваться обычными пользователями. При несанкционированном доступе к серверу, злоумышленник сможет установить скрипт, эмулирующий работу ssh-сервера и осуществить кражу паролей пользователей.

Приведенные выше правила можно осуществить следующими директивами в конфигурационном файле /etc/ssh/sshd_conf:

PermitRootLogin no – запрещает аутентификацию суперпользователя «root»,
PasswordAuthentication no – запрещает аутентификацию по паролю,
AllowUsers <users> – разрешает только перечисленных пользователей.

Результат функционирования правил можно увидеть на рис. 1.

```
210 Nov 30 01:17:37 [redacted] sshd[26202]: Received disconnect from 43.229.53.21: 11:  [preauth]
211 Nov 30 02:09:48 [redacted] sshd[26204]: User root from 43.229.53.21 not allowed because not listed in AllowUsers
212 Nov 30 02:09:48 [redacted] sshd[26204]: input_userauth_request: invalid user root [preauth]
213 Nov 30 02:09:48 [redacted] sshd[26204]: Received disconnect from 43.229.53.21: 11:  [preauth]
214 Nov 30 02:17:01 [redacted] CRON[26206]: pam_unix(cron:session): session opened for user root by (uid=0)
215 Nov 30 02:17:01 [redacted] CRON[26206]: pam_unix(cron:session): session closed for user root
216 Nov 30 02:23:45 [redacted] sshd[26209]: Did not receive identification string from 123.151.42.61
217 Nov 30 03:17:01 [redacted] CRON[26210]: pam_unix(cron:session): session opened for user root by (uid=0)
218 Nov 30 03:17:01 [redacted] CRON[26210]: pam_unix(cron:session): session closed for user root
219 Nov 30 04:16:48 [redacted] sshd[26213]: User root from 43.229.53.21 not allowed because not listed in AllowUsers
220 Nov 30 04:16:48 [redacted] sshd[26213]: input_userauth_request: invalid user root [preauth]
221 Nov 30 04:16:49 [redacted] sshd[26213]: Received disconnect from 43.229.53.21: 11:  [preauth]
222 Nov 30 04:17:01 [redacted] CRON[26215]: pam_unix(cron:session): session opened for user root by (uid=0)
223 Nov 30 04:17:01 [redacted] CRON[26215]: pam_unix(cron:session): session closed for user root
224 Nov 30 05:09:36 [redacted] sshd[26223]: User root from 43.229.53.21 not allowed because not listed in AllowUsers
225 Nov 30 05:09:36 [redacted] sshd[26223]: input_userauth_request: invalid user root [preauth]
226 Nov 30 05:09:36 [redacted] sshd[26223]: Received disconnect from 43.229.53.21: 11:  [preauth]
227 Nov 30 05:17:01 [redacted] CRON[26225]: pam_unix(cron:session): session opened for user root by (uid=0)
228 Nov 30 05:17:01 [redacted] CRON[26225]: pam_unix(cron:session): session closed for user root
229 Nov 30 06:16:12 [redacted] sshd[26228]: User root from 43.229.53.21 not allowed because not listed in AllowUsers
230 Nov 30 06:16:12 [redacted] sshd[26228]: input_userauth_request: invalid user root [preauth]
231 Nov 30 06:16:12 [redacted] sshd[26228]: Received disconnect from 43.229.53.21: 11:  [preauth]
232 Nov 30 06:17:01 [redacted] CRON[26230]: pam_unix(cron:session): session opened for user root by (uid=0)
233 Nov 30 06:17:01 [redacted] CRON[26230]: pam_unix(cron:session): session closed for user root
234 Nov 30 06:25:01 [redacted] CRON[26233]: pam_unix(cron:session): session opened for user root by (uid=0)
235 Nov 30 06:39:36 [redacted] CRON[26233]: pam_unix(cron:session): session closed for user root
236 Nov 30 07:01:10 [redacted] sshd[26415]: User root from 218.87.109.253 not allowed because not listed in AllowUsers
237 Nov 30 07:01:10 [redacted] sshd[26415]: input_userauth_request: invalid user root [preauth]
238 Nov 30 07:01:10 [redacted] sshd[26415]: Received disconnect from 218.87.109.253: 11:  [preauth]
239 Nov 30 07:17:01 [redacted] CRON[26417]: pam_unix(cron:session): session opened for user root by (uid=0)
240 Nov 30 07:17:01 [redacted] CRON[26417]: pam_unix(cron:session): session closed for user root
241 Nov 30 07:22:54 [redacted] sshd[26420]: User root from 43.229.53.21 not allowed because not listed in AllowUsers
242 Nov 30 07:22:54 [redacted] sshd[26420]: input_userauth_request: invalid user root [preauth]
243 Nov 30 07:22:54 [redacted] sshd[26420]: Received disconnect from 43.229.53.21: 11:  [preauth]
@
 NORMAL >> /var/log/auth.log                                    < messages < utf-8[unix] <  85% : 210:  1
```

Рис. 1. Результат функционирования правил

Эффективной мерой защиты может послужить использование утилиты fail2ban. Данная утилита позволяет установить лимит на количество попыток авторизации посредством ssh, при превышении которого создается политика блокировки ip-адреса на заданный промежуток времени. Для конфигурирования требуется указать следующие настройки в конфигурационном файле /var/fail2ban/jail.local:

bantime = 86400 – блокировка ip-адреса на 24 ч (в сек),
findtime = 120 – время, за которое считаются попытки,
maxretry = 3 – количество попыток,
[ssh] – включает мониторинг сервиса ssh,

enabled = true,
port = ssh,
filter = sshd,
logpath = /var/log/auth.log.

Результат работы утилиты можно увидеть на рис. 2.

```
r[illegible] me:~$ sudo iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
-N fail2ban-nginx-http-auth
-N fail2ban-ssh
-A INPUT -p tcp -m multiport --dports 22 -j fail2ban-ssh
-A fail2ban-ssh -s 80.153.99.58/32 -j REJECT --reject-with icmp-port-unreachable
-A fail2ban-ssh -s 188.116.52.14/32 -j REJECT --reject-with icmp-port-unreachable
-A fail2ban-ssh -s 40.121.48.153/32 -j REJECT --reject-with icmp-port-unreachable
-A fail2ban-ssh -j RETURN
r[illegible] me:~$
```

Рис. 2. Результат работы утилиты

Для защиты от сканирования может быть применена утилита psad. Утилита psad посредством мониторинга логов брандмауэра определяет попытки сканирования или атак на сервер. Принцип работы предельно прост: psad определяет, что с конкретного ip-адреса производится попытка сканирования сервера на открытые порты и по заранее заданному правилу осуществляет блокировку данного ip-адреса. Для конфигурирования требуется указать следующие настройки в файле конфигурации /etc/psad/psad.conf:

EMAIL_ADDRESSES address@domain.com,
HOSTNAME domain.com,
DANGER_LEVEL1 5,
DANGER_LEVEL2 15,
DANGER_LEVEL3 150,
DANGER_LEVEL4 1500,
DANGER_LEVEL5 10000,
PORT_RANGE_SCAN_THRESHOLD 1,
IPT_SYSLOG_FILE /var/log/syslog,
IGNORE_PORTS ports_or_range_to_ignore,
MIN_DANGER_LEVEL 1; # Controls psad logging and email alerts,
EMAIL_ALERT_DANGER_LEVEL 1; # Applies only for email alerts,
EMAIL_LIMIT 0,
ENABLE_AUTO_IDS Y,
AUTO_IDS_DANGER_LEVEL 5,
AUTO_BLOCK_TIMEOUT 3600.

При помощи утилиты nmap можно осуществить попытку сканирования сервера на наличие открытых портов:

sudo nmap -PN -sS server_domain_or_ip
Nmap scan report for server_domain_or_ip
Host is up (0.013s latency).
Not shown: 999 filtered ports
PORT STATE SERVICE
22/tcp open ssh
Nmap done: 1 IP address (1 host up) scanned in 6.84 seconds

Результатом служит наличие новых политик в брандмауэре iptables:

-A FORWARD -j PSAD_BLOCK_FORWARD
-A FORWARD -j LOG
-A OUTPUT -j PSAD_BLOCK_OUTPUT

Методика port knocking относится к так называемому принципу «безопасность через неясность». Принцип данной методики заключается в том, что нужный порт заблокирован политикой брандмауэра и для доступа к нему пользователь должен осуществлять доступ через

строго заданную цепочку портов. Для применения данной методики можно использовать утилиту knockd. Для конфигурирования требуется указать следующие настройки в файле конфигурации /etc/kockd.conf, а также сконфигурировать брандмауэр iptables. Для начала потребуется добавить следующие правила в iptables:

```
sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
sudo iptables -I INPUT 1 -i lo -j ACCEPT
sudo iptables -P INPUT DROP
```

Конфигурация knockd в файле /etc/knockd.conf:

```
[options]
UseSyslog
[openSSH]
sequence = 7000,8000,9000
seq_timeout = 5 command = /sbin/iptables -A INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
tcpflags = syn
[closeSSH] sequence = 9000,8000,7000
seq_timeout = 5
command = /sbin/iptables -D INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
tcpflags = syn
```

Результатом работающего сервиса knockd является недоступность подключения по порту 22 - ssh:

```
ssh <server-ip-address>
sh: connect to host server_ip_address port 22: Operation timed out
```

Чтобы получить доступ к серверу, требуется осуществлять подключение через цепочку портов:

```
knock 7000 8000 9000 && ssh <server-ip-address>
```

Для закрытия порта требуется всего лишь пройти цепочку в обратном порядке:

```
knock 9000 8000 7000
```

Заключение

Проведено исследование методик защиты удаленного подключения к серверу. Для тестирования использовался настроенный веб-сервер с доступом через сеть Интернет. В результате были получены работающие методики защиты удаленного подключения и навыки по их применению. Получена статистика, согласно которой в среднем за день осуществляется 35-40 попыток получения доступа перебором учетных данных. Данная статистика актуальна после применения приведенных выше методик (без использования методики port knocking).

PROTECTION OF REMOTE ACCESS TO SERVER AGAINST BRUTEFORCE ATTACK

R.M. HARBUL, D.D. TRAFIMENKA, F. ALIHANOV

Abstract

Remote access policy against bruteforce attacks was examined during research work of bruteforce attack characteristics and its risk factors. As research results remote access policy is developed and its efficiency against bruteforce attacks is shown.

Список литературы

1. *Олифер В.Г., Олифер Н.А.* Компьютерные сети. Принципы, технологии, протоколы. СПб., 2010.
2. *Таненбаум Э., Уэзеролл Д.* Компьютерные сети. СПб., 2012.
3. Iptables. [Электронный ресурс]. Режим доступа: http://ipset.netfilter.org/iptables.man.html.
4. Port Scan Attack Detector. [Электронный ресурс]. Режим доступа: https://cipherdyne.org/psad/docs/manpages/psad.html.
5. Fail2Ban. [Электронный ресурс]. Режим доступа: http://www.fail2ban.org/wiki/index.php/MANUAL_0_8.
6. OpenSSH. [Электронный ресурс]. Режим доступа: http://www.openssh.com/manual.html.
7. Port knocking manual pages. [Электронный ресурс]. Режим доступа: http://www.portknocking.org.