LINUX®
SERVER
BEZPIECZEŃSTWO I OCHRONA SIECI

CHRIS BINNIE

Helion

Tytuł oryginału: Linux Server Security: Hack and Defend

Tłumaczenie: Grzegorz Kowalczyk

ISBN: 978-83-283-3182-2

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.
The Second Edition was published by John Wiley & Sons, Inc. in 2010.

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Spis treści

ROZDZIAŁ

1

Czapka niewidka

Wyobraź sobie, że możesz całkowicie ukryć swój serwer w sieci Internet, ale tak, że nadal mógłbyś bez ograniczeń korzystać z jego zasobów. Takiego serwera mógłbyś używać na przykład jako bezpiecznego repozytorium plików (a lista innych zastosowań jest niemal nieograniczona).

Mając zdalny dostęp do konsoli serwera, mógłbyś instalować, uruchamiać i zatrzymywać usługi, z których chcesz korzystać. Wybór sposobu działania usług zależałby wyłącznie od Ciebie. W zależności od potrzeb mógłbyś na przykład uruchamiać wybrane usługi tylko na czas potrzebny do wykonania danego zadania i potem je zatrzymywać, czy uruchamiać inne usługi i pozostawiać je działające przez cały czas.

Możesz to osiągnąć poprzez zastosowanie techniki nazywanej *port knocking*. Aby ukryć serwer w sieci i spowodować, aby stał się „niewidzialny" dla innych hostów, możesz po prostu całkowicie zamknąć wszystkie porty sieciowe łączące serwer ze światem zewnętrznym. Jeżeli jednak „zapukasz" do takiego serwera w odpowiedni sposób (czyli inaczej mówiąc, gdy wyślesz do portów serwera odpowiednio przygotowaną sekwencję pakietów), oprogramowanie działające na serwerze rozpozna, że to Ty i otworzy dla Ciebie wcześniej wybrany port sieciowy (np. port 22 dla sesji SSH). W tym rozdziale pokażemy, jak możesz spowodować, aby Twój serwer był „niewidzialny" w sieci, oraz przedstawimy kilka przykładowych sposobów jego konfiguracji.

Wprowadzenie

Jeżeli spowodujesz, że Twój serwer będzie całkowicie niewidoczny w sieci Internet, to w najlepszym wypadku będziesz miał do dyspozycji swoją własną, prywatną maszynę, do której poza Tobą nikt inny nie będzie miał dostępu. W najgorszym wypadku, nawet jeżeli istnienie tego hosta nie będzie dla nikogo tajemnicą, poprzez ograniczenie czasu otwarcia poszczególnych portów znacząco zredukujesz płaszczyznę ataku, która mogłaby zostać wykorzystana przez potencjalnego napastnika.

Badanie otwartych portów

Zanim przejdziemy dalej, przyjrzymy się najpierw portom sieciowym serwera. Jeżeli kiedykolwiek używałeś skanerów sieciowych takich jak Nmap, to z pewnością spotkałeś się z nieco mylącymi raportami ze skanowania informującymi, że niektóre porty są zamknięte, podczas gdy w rzeczywistości tak nie jest. Nmap potrafi odróżnić, czy na porcie, który wydaje się być zamknięty, działa (nasłuchuje) jakaś usługa sieciowa, czy nie.

Nmap raportuje, że dany port jest zamknięty (ang. *closed*) w sytuacji, gdy nie działa na nim żadna usługa sieciowa, ale sam port wydaje się być otwarty lub przynajmniej potencjalnie dostępny. Jeżeli Nmap raportuje, że dany port jest filtrowany (ang. *filtered*), oznacza to, że skanowany host jest chroniony przez jakąś zaporę sieciową, która blokuje pakiety nadsyłane przez skaner z danego adresu IP (dotyczy to zwłaszcza pakietów `TCP RST`). Nmap rozpoznaje jeszcze trzy inne stany portów: niefiltrowany (ang. *unfiltered*), otwarty|filtrowany (ang. *open|filtered*) oraz zamknięty|filtrowany (ang. *closed|filtered*). Więcej szczegółowych informacji na temat stanów skanowanych portów znajdziesz na stronie *https://nmap.org/book/man-port-scanning-basics.html*.

Wprowadzanie w błąd skanera portów

Skoro już wiemy, w jaki sposób porty sieciowe serwera są „widziane" przez skanery portów, pokażemy teraz, jak można zmodyfikować odpowiedzi wysyłane przez serwer, tak aby wprowadzić w błąd nawet bardzo zaawansowane skanery portów wykorzystujące nieraz niezwykle wyrafinowane techniki skanowania. Jednym z najbardziej oczywistych narzędzi, których możemy użyć do tego celu, będzie działająca na poziomie jądra systemu i dysponująca ogromnym zestawem możliwości zapora sieciowa Netfilter, znana bardziej pod nazwą `iptables`.

A działa to tak. Najpierw musimy tak skonfigurować zaporę sieciową `iptables`, aby w przypadku odebrania skanujących pakietów TCP nasz system odpowiadał odesłaniem żądania `REJECT`. Pakiety innych protokołów możemy po prostu odrzucać za pomocą reguły `DROP`. Dzięki takiemu rozwiązaniu w raporcie wygenerowanym przez skaner (taki jak Nmap) porty naszego serwera będą oznaczone jako zamknięte (ang. *closed*), a nie filtrowane (ang. *filtered*). Na podstawie komentarzy i opinii użytkowników portali i forów dyskusyjnych poświęconych bezpieczeństwu systemów informatycznych możemy zaryzykować twierdzenie, że odpowiedź informująca o tym, iż skanowany port jest zamknięty (ang. *closed port*), jest w takiej sytuacji najbardziej pożądanym rozwiązaniem. W ten sposób bowiem nie pokazujemy otwarcie, że porty naszego serwera są blokowane przez zaporę sieciową ani że porty są otwarte i działają na nich określone usługi sieciowe (demony usług sieciowych).

Warto również zauważyć, że w normalnych okolicznościach skanowanie portu, który jest nieosiągalny, spowoduje wygenerowanie odpowiedzi *ICMP Port Unreachable*. Oczywiście w naszym przypadku nie chcemy, aby takie odpowiedzi były generowane, ponieważ wysłanie takiej odpowiedzi stanowiłoby jasną przesłankę informującą o tym, że za skanowanym portem kryje się serwer, co jednoznacznie ujawniłoby jego istnienie. Aby zamiast tego nasza zapora sieciowa spowodowała odesłanie odpowiedzi `REJECT`, możemy użyć następującej reguły: `reject-with tcp-reset`. Takie rozwiązanie powoduje, że dla zdalnego skanera nasza odpowiedź wygląda tak, jakby skanowany port był nieużywany i zamknięty, ale jednocześnie niefiltrowany.

Aby to zrobić, wystarczy dołączyć na końcu każdej reguły zapory `iptables` następujące polecenie:

```
-j REJECT-reject-with tcp-reset
```

Takie rozwiązanie zapewnia, że Twój system nie będzie niepotrzebnie ujawniał na zewnątrz żadnych informacji o swoim istnieniu.

Warto jednak zauważyć, że technika *port knocking*, którą omówimy już za chwilę, nie będzie korzystała z tej opcji konfiguracji reguł zapory `iptables`, ponieważ w naszym przypadku nie będziemy używać żadnych dodatkowych usług sieciowych poza SSH. Mimo to zdecydowaliśmy się na omówienie takiej konfiguracji, aby pokazać sposób, w jaki napastnik może skanować porty, oraz przedstawić możliwości zabezpieczenia innych usług sieciowych za pomocą reguły `reject-with tcp-reset`.

W sieci można znaleźć wiele dyskusji na temat wyższości (bądź nie) stosowania polecenia `DROP` zamiast `REJECT` w regułach zapory sieciowej `iptables`. Jeżeli jesteś bardzo zainteresowany tymi zagadnieniami, więcej szczegółowych informacji na ten temat znajdziesz na stronie *http://www.chiark.greenend.org.uk/~peterb/network/drop-vs-reject*.

Instalowanie pakietu knockd

W poprzednim podrozdziale przedstawiliśmy kilka podstawowych zagadnień stanowiących dobre przygotowanie do zastosowania port knockingu, możemy więc już pokazać, w jaki sposób należy zainstalować demona obsługującego tę technikę na swoim serwerze. Nieco później, już podczas konfiguracji demona, będziesz musiał się zastanowić, jakie usługi sieciowe chcesz ukryć przed innymi użytkownikami sieci Internet. Może to być dobra okazja na przykład do uruchomienia serwera WWW czy serwera poczty elektronicznej działającego przez kilka godzin na zupełnie niestandardowym porcie sieciowym.

Pakiety

Pokażemy teraz, w jaki sposób możesz zainstalować na swoim serwerze pakiet `knockd`, zawierający demona obsługującego port knocking. Proces instalacji tego pakietu będzie się różnił w zależności od tego, pod kontrolą jakiego systemu operacyjnego działa Twój serwer.

W przypadku systemów operacyjnych zbudowanych na bazie dystrybucji Debian pakiet demona `knockd` możesz zainstalować za pomocą poniższego polecenia:

```
# apt-get install knockd
```

W systemach opartych na dystrybucji Red Hat możesz użyć następującego polecenia:

```
# yum install knockd
```

Zdecydowana większość ustawień demona `knockd` jest przechowywana w pliku konfiguracyjnym. Na serwerach działających pod kontrolą systemu Debian Jessie jest to plik */etc/knockd.conf*. Na listingu 1.1 przedstawiamy przykładową zawartość tego pliku.

LISTING 1.1. Główny plik konfiguracyjny demona knockd. Domyślna sekwencja portów oraz opcji -I INPUT została zmodyfikowana

```
[options]
   UseSyslog
[openSSH]
   sequence = 6,1450,8156,22045,23501,24691
   seq_timeout = 5
   command = /sbin/iptables -I INPUT -s %IP% -p tcp-dport 22 -j
ACCEPT
   tcpflags = syn
[closeSSH]
   sequence = 3011,6145,7298
   seq_timeout = 5
   command = /sbin/iptables -D INPUT -s %IP% -p tcp-dport 22 -j
ACCEPT
   tcpflags = syn
```

Zmiana ustawień domyślnych

Na początku listingu 1.1 możemy zobaczyć sekcję opcji konfiguracji demona `knockd`. W pozostałych dwóch sekcjach możemy zdefiniować operacje, które zostaną wykonane wtedy, gdy `knockd` otworzy port SSH lub gdy dostęp do tego portu zostanie zablokowany. W obu sekcjach znajdziesz opcje `sequence`, które pozwalają na zdefiniowanie sekwencji „puknięć" wyzwalających akcje danej sekcji. Po zainstalowaniu pakietu `knockd` powinieneś jak najszybciej zmienić domyślne sekwencje „puknięć", tak aby uniknąć zagrożenia dla bezpieczeństwa serwera. Domyślnie dostęp do SSH otwiera wysłanie pakietów kolejno na porty 7000, 8000 i 9000, a zamyka wysłanie pakietów na porty 9000, 8000 i 7000. Jak łatwo zauważyć, w naszym przykładzie sekwencja portów otwierająca dostęp do SSH została znacząco rozbudowana, tak aby zredukować możliwość niezamierzonego otwarcia dostępu do portu poprzez przypadkowy skan portów.

Po zmianie dowolnych ustawień konfiguracyjnych powinieneś zrestartować demona `knockd`. W systemach operacyjnych wykorzystujących menedżera `systemd` możesz to zrobić za pomocą następującego polecenia:

```
# systemctl restart knockd.service
```

Po zainstalowaniu demona `knockd` powinieneś zapoznać się z jego dokumentacją. W dystrybucji Debian Jessie możesz rozpocząć od pliku *README*, który znajdziesz w katalogu */usr/share/doc/knockd/*.

We wspomnianym pliku *README* znajdziesz szczegółowy opis sposobu działania pakietu. Demon `knockd` wykorzystuje bibliotekę `libpcap`, która używana jest również przez wiele innych pakietów, takich jak na przykład `tcpdump`, `ngrep` czy `iftop` (pakiet pozwalający na przechwytywanie i analizę połączeń sieciowych w czasie rzeczywistym). Dzięki kilku sprytnym rozwiązaniom demon `knockd` nie musi być nawet powiązany z monitorowanymi portami, a mimo to umożliwia skryte nasłuchiwanie napływającego do nich ruchu sieciowego.

Zmiana lokalizacji plików

Zdarzenia takie, jak nawiązanie połączenia, rozłączenie połączenia czy błędy, są logowane bezpośrednio przez system operacyjny i mogą być zapisywane w plikach */var/log/messages* lub */var/log/syslog*. Jeżeli nie chcesz, aby informacje z demona `knockd` zostały „zagubione" pod stosem innych komunikatów zapisywanych w logach systemowych, lub po prostu nie lubisz tracić czasu na mozolne „wygrzebywanie" interesujących Cię informacji spośród tysięcy innych rekordów, możesz utworzyć swój własny niestandardowy plik logu. Osobiście preferuję takie właśnie rozwiązanie, ponieważ znakomicie ułatwia ono analizowanie historii działania demona i nawiązywanych połączeń, a co więcej, dzięki temu do analizy logu możemy użyć różnego rodzaju zautomatyzowanych narzędzi czy własnych skryptów. Aby utworzyć własny log, powinieneś w pliku konfiguracyjnym demona umieścić następujące polecenie:

```
[options]
   LogFile = /var/log/portknocking.log
```

Zmiana lokalizacji pliku logu jest powszechnie stosowanym rozwiązaniem, ale warto również wiedzieć, że można zmienić miejsce, w którym po uruchomieniu demona `knockd` zapisywany będzie plik `pid` (ang. *Process ID*). Lokalizację tego pliku można zmienić, dodając następujący wpis w sekcji `[options]`:

```
[options]
   PidFile = /var/tmp/run/file
```

Niektóre opcje konfiguracyjne

Wiemy już, jaką strukturę ma główny plik konfiguracyjny demona knockd, możemy zatem przystąpić teraz do dostosowywania go do własnych potrzeb. W tej sekcji poznasz różne opcje konfiguracji i dowiesz się, jaką rolę w konfiguracji Twojego serwera odgrywają limity czasu działania różnych opcji.

Uruchamianie usługi

Nie wpadaj w panikę, jeżeli na ekranie pojawi się komunikat informujący o tym, że demon knockd jest zablokowany. Jest to zupełnie normalne zachowanie demona, dzięki któremu przed zakończeniem konfiguracji knockd nie wprowadza do reguł zapory iptables żadnych niepożądanych zmian.

W dystrybucji Debian Jessie komunikat o błędzie prosi Cię o zmianę w pliku */etc/default/knockd* wartości parametru START_KNOCKD na 1:

```
START_KNOCKD=1
```

Oczywiście powinieneś to zrobić dopiero po uważnym sprawdzeniu konfiguracji oraz upewnieniu się, że w razie wystąpienia jakichkolwiek problemów z połączeniem sieciowym nadal będziesz miał dostęp do serwera.

Zmiana domyślnego interfejsu sieciowego

Po skonfigurowaniu żądanej sekwencji portów (lub inaczej mówiąc, sekwencji „puknięć") możesz rozpocząć modyfikowanie innych parametrów. W pliku konfiguracyjnym */etc/default/knockd* znajdziesz między innymi opcję KNOCKD_OPTS, która pozwala na zmianę interfejsu sieciowego, na którym będzie nasłuchiwał demon knockd. Przykład modyfikacji ustawień tego parametru został przedstawiony poniżej:

```
KNOCKD_OPTS="-i eth1"
```

Po zmianie ustawień powinieneś zrestartować demona knockd. W systemach wykorzystujących menedżera systemd możesz to zrobić za pomocą następującego polecenia:

```
# systemctl restart knockd
```

Rodzaje pakietów i limity czasu

W pliku */etc/knockd.conf* znajdziesz kilka parametrów, których ustawienia mają wpływ na to, w jaki sposób klienty będą się łączyć z Twoim serwerem. Aby przypomnieć sobie, jak wyglądała zawartość tego pliku, wróć na chwilę do listingu 1.1, a następnie w sekcji [openSSH] dodaj następujące opcje:

```
[openSSH]
   tcpflags = syn
   seq_timeout = 10
   cmd_timeout = 15
```

Opcja tcpflags oznacza, że knockd będzie akceptował pakiety TCP tylko określonego tutaj typu — w naszym przypadku będą to pakiety TCP z ustawioną flagą SYN. Możesz również używać takich flag jak fin, syn, rst, psh, ack oraz urg. Jeżeli pakiet przychodzący do portu nie będzie miał ustawionej odpowiedniej flagi, to zostanie po prostu przez demona knockd zignorowany. Pamiętaj jednak, że nie jest to normalne zachowanie demona knockd. W domyślnej konfiguracji otrzymanie nieprawidłowego pakietu spowoduje przerwanie i odrzucenie całej sekwencji otrzymanych pakietów, co oznacza, że aby uzyskać dostęp

do serwera, klient musi ponownie przesłać całą sekwencję. Jeżeli akceptowane mają być pakiety z różnymi flagami, możesz podać listę dozwolonych flag, oddzielając je od siebie przecinkami. Co ciekawe, zgodnie z informacjami w logu zmian (począwszy od wersji 0.5 pakietu) w konfiguracji demona można za pomocą znaku wykrzyknika wykluczać wybrane typy pakietów, np.:`!ack`.

Ale powróćmy do innych opcji w naszym przykładowym pliku konfiguracyjnym. Z pewnością zauważyłeś, że w pliku przedstawionym na listingu 1.1 znajduje się opcja `seq_timeout`, która określa maksymalny czas odebrania całej sekwencji „puknięć". Ponieważ w naszym przypadku otwierająca sekwencja pakietów jest znacznie dłuższa niż sekwencja domyślna, warto pomyśleć o zwiększeniu wartości opcji `seq_timeout` z 5 na 10. Takie zwiększenie może być konieczne, ponieważ w przypadku klientów działających na niezbyt szybkich połączeniach sieciowych (na przykład ze smartfonów) mogą występować przekroczenia dozwolonego czasu przesyłania sekwencji pakietów.

Na koniec warto wspomnieć również o opcji `cmd_timeout`, która odnosi się do tego, co się wydarzy po tym, gdy `knockd` odbierze poprawną sekwencję „puknięć". Gdy poprawność sekwencji zostanie potwierdzona, `knockd` uruchomi polecenie umieszczone w opcji `start_command` (zobacz listing 1.1). Jeżeli opcja `cmd_timeout` jest obecna, to po uruchomieniu polecenia `start_command` demon czeka przez czas podany w opcji `cmd_timeout`, a następnie automatycznie uruchamia polecenie zdefiniowane w opcji `stop_command`.

Jest to preferowany sposób pracy z sesjami SSH. Po wysłaniu odpowiedniej sekwencji „puknięć" nawiązujemy połączenie SSH z serwerem i po upływie niewielkiego czasu ponownie zamykamy port. W takiej sytuacji nie powinieneś mieć żadnych kłopotów z pracą w rozpoczętej sesji SSH, ale wszystkie próby nawiązania nowego połączenia SSH będą musiały ponownie przejść cały proces „uwierzytelniania" za pomocą odpowiedniej sekwencji pakietów. Możesz to sobie wyobrazić jako swego rodzaju zamykanie drzwi po wejściu do domu. Po nawiązaniu połączenia i zamknięciu portu Twój serwer ponownie staje się niewidzialny w sieci, a jedynym czynnikiem wskazującym na jego istnienie jest ruch sieciowy powiązany z Twoją sesją SSH.

Testowanie zainstalowanego pakietu

Ponieważ w grę wchodzi tutaj bezpieczeństwo Twojego serwera, po zainstalowaniu pakietu `knockd` powinieneś wykonać szereg testów mających na celu upewnienie się, że wszystko działa zgodnie z Twoimi oczekiwaniami. Najlepiej będzie, jeżeli w tym celu skorzystasz z innej maszyny, spełniającej w testach rolę klienta. Aby mieć całkowitą pewność, że `knockd` poprawnie otwiera i zamyka odpowiednie porty, zazwyczaj dokonuję prób połączenia z klienta, który posiada zupełnie inny adres IP. Jeżeli nie masz dostępu do komputera z innym adresem IP, możesz skorzystać z laptopa podłączonego do sieci bezprzewodowej lub nawet z klienta działającego na Twoim smartfonie.

Klienty port knockingu

Do wysłania odpowiedniej sekwencji pakietów i nawiązania połączenia SSH możesz użyć jednego z wielu dostępnych klientów. W razie potrzeby do ręcznego wysłania odpowiedniej sekwencji pakietów możesz nawet użyć takich programów jak Nmap, `netcat` czy `telnet`. Dokumentacja pakietu `knockd` wymienia również programy takie, jak `hping`, `sendip` i `packit`. W tym podrozdziale przyjrzymy się poleceniu `knock`, które jest instalowane jako część pakietu `knockd`.

Jeżeli użyłeś konfiguracji przedstawionej w sekcji openSSH na listingu 1.1, możesz skorzystać z prostego polecenia knock, którego składnię przedstawiamy poniżej:

```
# knock [opcje] <host> <port[:protokół]> <port[:protokół]> <port[:protokół]>
```

Polecenie knock dla konfiguracji przedstawionej na listingu 1.1 będzie wyglądało więc następująco:

```
# knock 11.11.11.11 6:tcp 1450:tcp 8156:tcp 22045:tcp 23501:tcp 24691:tcp
```

W naszym przykładzie adres IP serwera docelowego to 11.11.11.11. Jeżeli chcesz, możesz oczywiście demona knockd skonfigurować tak, aby akceptował sekwencje pakietów różnie wykorzystujących dowolną kombinację portów TCP i UDP. W takiej sytuacji odpowiednie polecenie knock mogłoby wyglądać na przykład tak:

```
# knock 11.11.11.11 6:tcp 1450:udp 8156:udp 22045:tcp 23501:udp 24691:tcp
```

Jeżeli w sekwencji „puknięć" używasz wyłącznie portów UDP, możesz uprościć składnię polecenia knock, dodając w wierszu polecenia opcję -u i pomijając deskryptory protokołów przy numerach portów w dalszej części polecenia. Polecenie knock, które wykorzystuje tylko porty UDP w skróconej wersji, może więc wyglądać następująco:

```
# knock -u 11 22 33 44 55
```

A jak skonfigurować demona knockd do sekwencji wykorzystującej porty UDP i TCP? Aby to zrobić, powinieneś w pliku konfiguracyjnym demona odszukać sekcję openSSH i w opcji sequence wpisać odpowiednią sekwencję portów i protokołów, tak jak zostało to pokazane poniżej:

```
[openSSH]
   sequence = 6:tcp 1450:udp 8156:udp 22045:tcp 23501:udp 24691:tcp
```

Ukrywanie serwera w sieci

Jeżeli jesteś już pewien, że demon knockd został skonfigurowany poprawnie i wszystko działa zgodnie z oczekiwaniami, możesz rozpocząć ukrywanie serwera w sieci, tak aby przestał być widoczny dla potencjalnych napastników (i przy okazji wszystkich innych, nieuprawnionych użytkowników w sieci). Potencjalny napastnik może znać adres IP Twojego serwera, a w szczególnych okolicznościach może nawet mieć możliwość obserwacji ruchu przychodzącego i wychodzącego z tego adresu IP (gdy na przykład napastnik jest zatrudniony u dostawcy ISP, do którego sieci podłączony jest Twój serwer).

Jeżeli wszystko pójdzie zgodnie z oczekiwaniami, po ukryciu serwer nie powinien być widoczny dla użytkowników sieci Internet. Jeżeli tak nie jest, powinieneś sprawdzić ustawienia zapory sieciowej. W moim przypadku po wykonaniu poleceń opisanych powyżej Nmap zaczął raportować wszystkie porty jako zamknięte.

Testowanie reguł zapory iptables

Jak już wspominałem wcześniej, w naszych przykładach będziemy używać popularnej i sprawdzonej zapory sieciowej iptables. W idealnych warunkach przed rozpoczęciem modyfikowania reguł zapory sieciowej powinieneś mieć fizyczny dostęp do serwera, tak aby można było dokonać niezbędnych poprawek, jeżeli coś nie pójdzie zgodnie z oczekiwaniami. Jeżeli nie masz fizycznego dostępu do serwera, powinieneś zapewnić sobie inny, niezależny sposób dostępu do serwera, taki jak dostęp za pośrednictwem konsoli maszyny wirtualnej, dostęp do innego interfejsu sieciowego, za pomocą którego możesz się zalogować do serwera, lub nawet dostęp za pomocą modemu telefonicznego podłączonego do serwera. Pamiętaj, że jeżeli gruntownie nie przetestujesz konfiguracji zapory sieciowej i demona knockd w środowisku

deweloperskim, to istnieje bardzo duże niebezpieczeństwo, że popełnisz jakiś błąd i narobisz sobie kłopotów. Choć do tej pory postawiłem i skonfigurowałem całkiem sporo serwerów wykorzystujących *port knocking*, to nadal zdarzają mi się wpadki blokujące dostęp do serwera i od czasu do czasu muszę się ratować za pomocą alternatywnych metod nawiązania połączenia.

Mając na względzie powyższe ostrzeżenie, możemy rozpocząć modyfikowanie reguł zapory `iptables`. Pamiętaj o zachowaniu ostrożności podczas integrowania poleceń przedstawionych poniżej z istniejącymi regułami zapory. Dobrym rozwiązaniem może być na przykład wykonanie kopii zapasowej istniejącej konfiguracji zapory i następnie zmodyfikowanie odpowiednich reguł. Najpierw musisz się upewnić, że Twój serwer może komunikować się sam ze sobą za pomocą interfejsu `localhost`. Aby to zrobić, użyjemy następującej reguły zapory `iptables`:

```
# iptables -A INPUT -s 127.0.0.0/8 -j ACCEPT
```

Następnie musimy się upewnić, że wszystkie istniejące połączenia będą potwierdzone i pozostaną aktywne, tak jak to zostało pokazane poniżej:

```
# iptables -A INPUT -m conntrack-ctstate ESTABLISHED,RELATED -j ACCEPT
```

Do śledzenia nawiązanych połączeń używamy opcji `conntrack`, dzięki czemu po pomyślnym nawiązaniu sesji połączenia pozostaną aktywne i w pełni funkcjonalne. Zanim przejdziemy do kolejnych poleceń, przyjmiemy założenie, że do nawiązania sesji SSH na Twoim serwerze będziemy otwierać tylko port 22. Aby to zrobić, do pliku konfiguracyjnego demona `knockd`, przedstawionego na listingu 1.1, powinieneś dodać następujące polecenie:

```
command = /sbin/iptables -I INPUT -s %IP% -p tcp-dport 22 -j ACCEPT
```

Przyjrzyj się uważnie przedstawionemu poleceniu. Jeżeli zamiast opcji `-I` użyjesz opcji `-A INPUT` (ang. *append*), wówczas zapora `iptables` odetnie Ci dostęp do serwera. Aby tego uniknąć, musisz użyć opcji `-I INPUT` (ang. *insert*), dzięki czemu nowa reguła zostanie wstawiona do zbioru reguł jako pierwsza i będzie miała pierwszeństwo nad innymi regułami.

Z pewnością zastanawiasz się, czym jest zmienna `%IP%`. Pakiet `knockd` jest na tyle sprytny, że zamiast tej zmiennej potrafi automatycznie podstawić w polu `-s` odpowiedni adres IP hosta nawiązującego połączenie.

Od tej chwili musisz zachować szczególną ostrożność. Pamiętaj, że jeżeli coś pójdzie nie tak i stracisz dostęp do serwera, to nie będziesz mógł tego w prosty sposób cofnąć, dlatego przed wykonaniem kolejnego polecenia upewnij się, że masz sprawdzony sposób alternatywnego dostępu do serwera, tak na wszelki wypadek. Aby zablokować cały ruch przychodzący do Twojego serwera, powinieneś wykonać następujące polecenie:

```
# iptables -A INPUT -j DROP
```

Stało się. Jeżeli teraz za pomocą polecenia przedstawionego poniżej sprawdzisz reguły zapory sieciowej, nie powinieneś zobaczyć żadnych śladów wskazujących na to, że port 22 (SSH) może być otwarty:

```
# iptables -nvL
```

Regułę otwierającą port 22 możesz zobaczyć dopiero po pomyślnym nawiązaniu połączenia z serwerem (aczkolwiek będzie ona widoczna tylko przez czas ustawiony w opcji `cmd_timeout`, o ile oczywiście korzystasz z tej opcji).

Jeżeli napotkałeś jakieś problemy, przejdź do sekcji „Diagnozowanie i usuwanie problemów” w dalszej części tego rozdziału i spróbuj zwiększyć poziom logowania komunikatów systemowych, co może ułatwić Ci diagnozowanie przyczyn niepowodzenia. Jeżeli jednak wszystko pójdzie zgodnie z oczekiwaniami, to od tej chwili Twój serwer nie powinien być już widoczny w sieci, a skaner portów powinien raportować wszystkie porty jako zamknięte, tak jak to zostało pokazane na rysunku 1.1.

```
Starting NMap 6.47 ( http://nmap.org ) at 2015-11-26 17:33 GMT
Note: Host seems down. If it is really up, but blocking our ping problems, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 3.18 seconds
```

RYSUNEK 1.1. Nmap pokazuje, że pod tym adresem IP nie ma żadnego hosta

Zapisywanie reguł zapory iptables

Aby się upewnić, że nowe reguły zapory sieciowej `iptables` nie zostaną utracone po restarcie serwera, powinieneś zainstalować pakiet `iptables-persistent`. Na systemach opartych na dystrybucji Debian możesz to zrobić za pomocą następującego polecenia:

```
# apt-get install iptables-persistent
```

Po zainstalowaniu pakietu możesz zapisywać reguły zapory sieciowej `iptables` za pomocą takiego polecenia:

```
# /etc/init.d/iptables-persistent save
```

Aby powrócić do zapisanej wcześniej konfiguracji reguł zapory, możesz wykonać następujące polecenie:

```
# /etc/init.d/iptables-persistent reload
```

Na systemach opartych na dystrybucji Red Hat (przed wprowadzeniem menedżera `systemd`) możesz użyć następującego polecenia:

```
# /sbin/service iptables save
```

Aby przywrócić zapisane wcześniej reguły zapory, możesz wykonać polecenie przedstawione poniżej:

```
# /sbin/service iptables reload
```

Aby polecenia przedstawione powyżej działały na systemach Red Hat wykorzystujących menedżera `systemd`, powinieneś najpierw zainstalować pakiet `iptables-services`. Aby to zrobić, możesz użyć następującego polecenia:

```
# yum install iptables-services
```

Inne zagadnienia

Istnieje jeszcze wiele innych zagadnień dotyczących port knockingu, o których warto wspomnieć. Niektóre z nich omówimy w kolejnych podrozdziałach.

Klienty działające na smartfonach

Na smartfonach działających pod kontrolą systemu Android moim ulubionym klientem SSH jest aplikacja o nazwie JuiceSSH (zobacz *https://juicessh.com/*). Program pozwala na zainstalowanie dodatkowej wtyczki umożliwiającej skonfigurowanie odpowiedniej sekwencji pakietów, która będzie wykorzystywana jako część procesu uwierzytelniania SSH. Istnienie takiej aplikacji dla platformy mobilnej oznacza, że nie będziesz już miał żadnej wymówki do nieużywania port knockingu, nawet jeżeli często podróżujesz i nie zabierasz ze sobą laptopa.

Diagnozowanie i usuwanie problemów

Jeżeli wykonasz polecenie `tail -f <nazwa_pliku>`, gdzie *<nazwa_pliku>* to nazwa pliku dziennika (logu) demona port knockingu, to przekonasz się, że zawiera on szereg różnych wpisów. Znajdziesz tam między innymi informacje o tym, czy zdalny klient „pukał" do właściwych portów, a przede wszystkim, czy użył poprawnej sekwencji „puknięć".

Jeżeli chcesz zwiększyć poziom dokładności logowania aktywności demona port knockingu, możesz włączyć opcję debugowania. Aby to zrobić, powinieneś otworzyć w swoim ulubionym edytorze tekstu plik konfiguracyjny */etc/init/d/knockd*, poszukać w nim wiersza `OPTIONS`, a następnie do ciągu znaków będących jej wartością dodać opcję `-D`, tak jak to zostało pokazane poniżej:

```
OPTIONS="-d -D"
```

Po zakończeniu diagnozowania i usunięciu problemu z funkcjonowaniem demona `knockd` powinieneś wyłączyć opcję rozszerzonego logowania, dzięki czemu będziesz mógł uniknąć niepotrzebnego zapełniania miejsca na dysku przez rosnący plik logu. Opcja `-d` powoduje, że pakiet `knockd` jest uruchamiany jako demon. W normalnych warunkach jest to domyślny tryb działania pakietu `knockd`, dlatego w większości przypadków powinieneś pozostawić tę opcję bez zmian.

Wróćmy teraz na chwilę do klienta. Tutaj również w razie potrzeby możesz rozszerzyć zakres logowanych informacji, dodając w konfiguracji klienta opcję `-v`. Takie rozwiązanie (w połączeniu z włączonym trybem debugowania aktywności demona `knockd` na serwerze) powinno Ci dać całkiem pokaźny zestaw informacji, pozwalający na szybkie i precyzyjne zlokalizowanie i usunięcie problemu z funkcjonowaniem demona i nawiązywaniem połączenia zarówno po stronie serwera, jak i klienta.

Bezpieczeństwo

Jeżeli chcesz zminimalizować ilość publicznie dostępnych informacji o Twoim serwerze, powinieneś się postarać, aby Twój dostawca internetu (ISP) nie publikował informacji DNS i revDNS o adresie IP Twojego serwera. Jeżeli chcesz, aby serwer był naprawdę „niewidzialny" w sieci, jego adres IP powinien wyglądać na nieużywany.

Nawet jeżeli udostępniasz na swoim serwerze jakieś usługi publiczne, takie jak HTTP, powinieneś zawsze pamiętać o ukrywaniu informacji o wersjach demonów usług, których używasz. W przypadku serwera Apache, czyli najpopularniejszego na świecie serwera WWW, najprostszym sposobem ukrycia jego wersji jest zmiana w ustawieniach konfiguracyjnych opcji `ServerTokens` na `Prod` oraz ustawienie opcji `ServerSignature` na `Off`. Są to proste, ale jednocześnie skuteczne zmiany w konfiguracji serwera, które bardzo często powodują, że zautomatyzowane ataki sieciowe po prostu będą ignorowały i pomijały Twój serwer, ponieważ numer wersji Twojego serwera Apache nie zostanie znaleziony w bazie numerów wersji podatnych na taki czy inny atak (dotyczy to zwłaszcza ataków z wykorzystaniem exploitów typu *zero-day*).

Kolejne zagadnienie jest dosyć szczegółowo opisane w dokumentacji pakietu `knockd`. Znajdziesz tam informację o tym, że jeżeli podczas uruchamiania demona używasz opcji `-l` lub `--lookup`, która pozwala na rozwiązywanie nazw hostów i zapisywanie ich w logach, to takie rozwiązanie może stanowić nieco większe zagrożenie bezpieczeństwa Twojego serwera. Jeżeli korzystasz z tej opcji, to istnieje pewne niebezpieczeństwo, że niektóre informacje mogą zostać przechwycone przez potencjalnego napastnika. Mając na przykład możliwość obserwowania ruchu DNS wychodzącego z Twojego serwera, napastnik może z dużą dokładnością określić pierwszy port sekwencji otwierającej.

Sekwencje efemeryczne

Jeżeli chcesz naprawdę dobrze zabezpieczyć dostęp do serwera, możesz skorzystać z nieco innego sposobu definiowania sekwencji otwierających dostęp do serwera. Pakiet `knockd` pozwala na zdefiniowanie listy sekwencji „puknięć", które natychmiast po użyciu tracą ważność. Aby włączyć takie jednorazowe sekwencje, powinieneś powrócić do głównego pliku konfiguracyjnego demona `knockd`, który prezentowaliśmy na listingu 1.1, a następnie w sekcjach `open` i `close` dodać opcję przedstawioną poniżej:

```
[openSSH]
   One_Time_Sequences = /usr/local/etc/portknocking_codes
```

Jeżeli z pliku konfiguracyjnego, przedstawionego na listingu 1.1, usuniesz wiersz z opcją `sequence` i zastąpisz ją kodem przedstawionym powyżej, demon `knockd` będzie pobierał odpowiednie sekwencje „puknięć" z pliku zdefiniowanego w opcji `One_Time_Sequences`.

Sposób, w jaki `knockd` korzysta z sekwencji jednorazowych, jest nieco nietypowy. Po uruchomieniu demon pobiera z pliku pierwszą sekwencję i oczekuje na połączenie. Gdy klient prześle poprawną sekwencję i nawiąże połączenie z serwerem, demon `knockd` umieszcza na początku wiersza zawierającego tę sekwencję znak komentarza (zazwyczaj jest to znak #) i pobiera nową sekwencję z kolejnego wiersza. Wraz z nadejściem następnego poprawnego połączenia cała procedura się powtarza.

W dokumentacji pakietu znajdziemy zalecenie mówiące, że na początku każdego wiersza sekwencji powinieneś pozostawić jedną spację — w przeciwnym razie dodawanie znaku # mogłoby spowodować nadpisanie początku sekwencji i zablokowanie dostępu.

W pliku sekwencji w każdym wierszu powinieneś umieszczać tylko jedną sekwencję „puknięć". Składnia i format definicji sekwencji są takie same jak składnia sekwencji w opcji `sequence` w głównym pliku konfiguracyjnym demona `knockd`.

W dokumentacji pakietu znajdziesz również informację o tym, że w pliku sekwencji możesz zapisywać swoje własne komentarze, umieszczając je w wierszach rozpoczynających się od znaku #. Pamiętaj jednak, że próba edytowania pliku sekwencji po uruchomieniu demona `knockd` może się zakończyć w zupełnie nieprzewidziany sposób, a w najgorszym razie może nawet spowodować zablokowanie dostępu do serwera.

Gdy zapoznasz się już z dokumentacją i specyfiką działania demona `knockd`, możesz przystąpić do jego testowania. W fazie prób i testów jako sekwencje „puknięć" możesz wykorzystywać numery telefonów lub inne łatwe do zapamiętania ciągi liczb. Jako sekwencji otwierających możesz na przykład używać rotacyjnie kilku dobrze Ci znanych numerów telefonów, gdzie poszczególne grupy cyfr będą się przekładały na numery portów używanych w danej sekwencji.

Podsumowanie

W tym rozdziale oprócz sposobów ukrywania swojego serwera w sieci dowiedziałeś się, jak Twój serwer jest „widziany" przez napastnika przed rozpoczęciem ataku. Jeżeli chcesz niemal całkowicie ukryć swój serwer, korzystając z techniki port knockingu, powinieneś również pomyśleć o publicznych źródłach informacji, takich jak wpisy revDNS, które mogą poinformować potencjalnego napastnika o tym, że dany adres IP jest używany. Możesz również zastanowić się nad ukryciem serwera przy użyciu translacji adresów (NAT). Innym rozwiązaniem może być dynamiczne zmienianie co pewien czas adresu IP serwera

i informowanie uprawnionych użytkowników o nowym adresie za pomocą uzgodnionej wcześniej nazwy hosta, publikowanej bez rozgłosu na serwerze DNS wybranej (nie rzucającej się w oczy) domeny internetowej.

Zabezpieczanie serwerów to temat bardzo złożony i wielopłaszczyznowy, ale mam nadzieję, że w tym rozdziale udało mi się przedstawić wystarczająco wiele zagadnień, abyś wiedział, jakie informacje o serwerze mogą wydostawać się na zewnątrz, jak napastnik może użyć ich do przeprowadzenia ataku i w jaki sposób (korzystając z techniki port knockingu) możesz ukryć w sieci swój serwer.

Skorowidz

I

K

L

Ł

M

N

O

P

R

S

T

U

W

Z

UWAŻAJ! WROGOWIE W SIECI ZAWSZE SĄ BLISKO!

W sieci trwa wyścig zbrojeń. Na systemy internetowe czyha cała armia napastników o różnych umiejętnościach i intencjach. Niektórzy z nich mogą poszczycić się wysoką skutecznością, a efekty ich działań są tragiczne. Niezależnie od tego, jak nowocześnie i doskonale zabezpieczysz usługę sieciową, prędzej czy później stanie się ona podatna na ataki. Jedyne, co możesz zrobić, to cały czas pozostawać w pełnej gotowości, odpowiednio wcześnie wykrywać próby ataku i sukcesywnie je neutralizować.

Niniejsza książka jest przeznaczona dla administratorów linuksowych serwerów usług sieciowych, jednak może z niej korzystać każdy, kto interesuje się bezpieczeństwem systemów internetowych. Przedstawiono tu m.in. sytuacje naruszeń bezpieczeństwa oraz sposoby ich wykrywania i neutralizacji. Można je wykorzystać w różnych dystrybucjach systemu Linux i innych systemach z rodziny Unix. Pokazano również szereg narzędzi i technik stosowanych w atakach sieciowych oraz wskazano, jak administrator może je wykorzystywać. Jednym słowem, masz w ręku znakomite i aktualne kompendium wiedzy, które ułatwi Ci utrzymanie odpowiedniego poziomu bezpieczeństwa serwera.

Najciekawsze zagadnienia:

- skanowanie portów i ukrywanie serwera w sieci
- integralność systemu plików
- ataki lustrzane
- dzienniki systemowe i skrypty w służbie bezpieczeństwa
- złośliwe oprogramowanie i łamanie haseł za pomocą narzędzia Hashcat
- ataki SQL injection i wykorzystanie luk w SQL

CHRIS BINNIE jest konsultantem technicznym. Ma ponad 20-letnie doświadczenie w pracy z systemami Linux. Jest autorem wielu artykułów publikowanych w takich czasopismach, jak „Linux Magazine" czy „ADMIN Magazine". Jednym z jego osiągnięć było zaprojektowanie i zbudowanie platformy streamingowej, za której pośrednictwem sygnał wideo w rozdzielczości HD był przesyłany do 77 krajów świata. Zdobył też ogromne doświadczenie w instalowaniu i konfigurowaniu serwerów linuksowych dla instytucji bankowych i rządowych.

Helion
księgarnia internetowa
http://helion.pl
zamówienia telefoniczne
0 801 339900
0 601 339900
Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
http://helion.pl
Sprawdź najnowsze promocje:
http://helion.pl/promocje
Książki najchętniej czytane:
http://helion.pl/bestsellery
Zamów informacje o nowościach:
http://helion.pl/nowosci
Informatyka w najlepszym wydaniu
ISBN 978-83-283-3182-2
9 788328 331822
cena: 39,90 zł