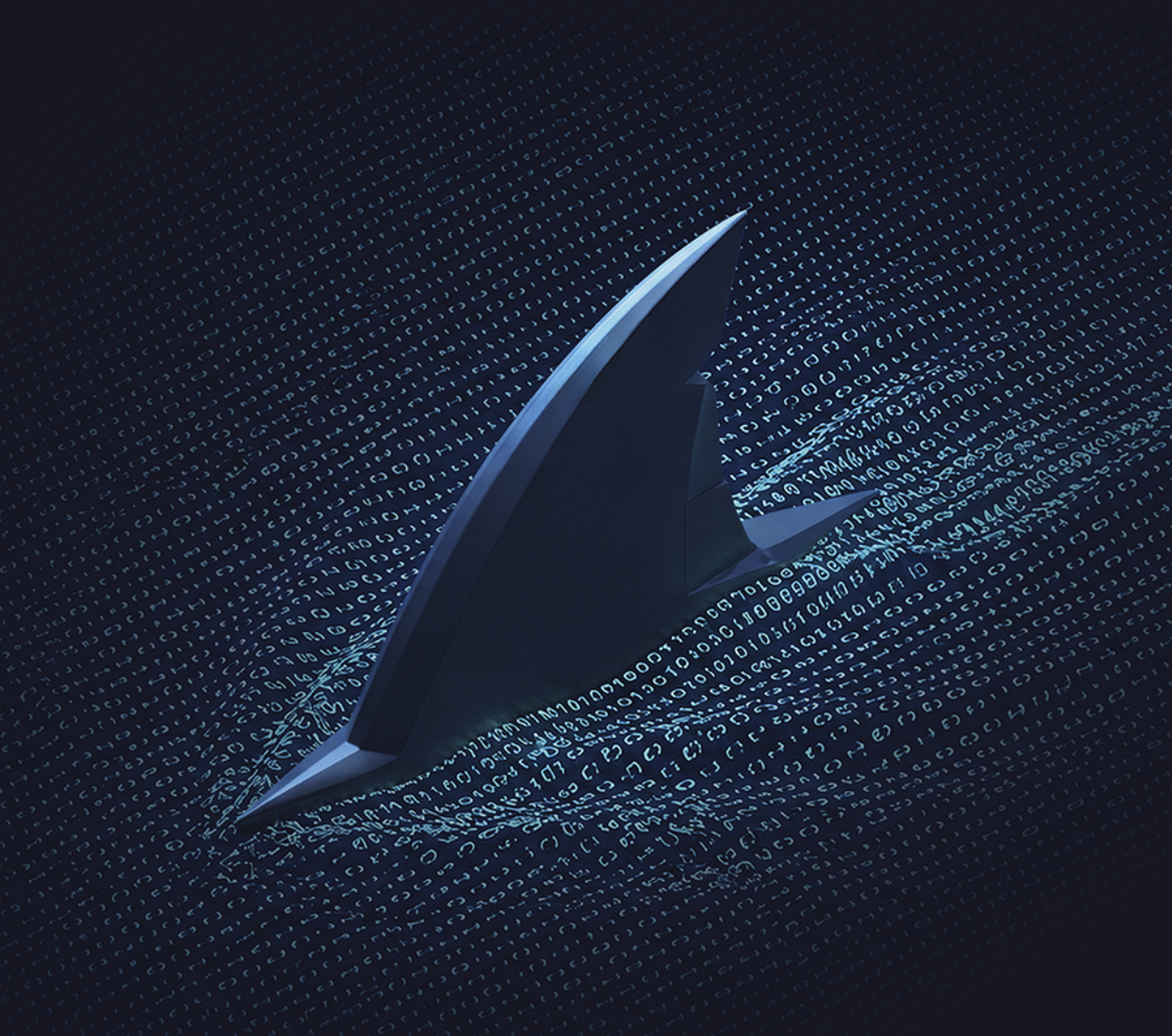




Adam Józefiok

WIRESHARK

Analiza ruchu sieciowego
i wykrywanie włamań

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

Grafika na okładce została wykorzystana za zgodą AdobeStock.com.

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

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

ISBN: 978-83-289-2639-4

Copyright © Helion S.A. 2026

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

Wprowadzenie	9
ROZDZIAŁ 1. Podstawy programu Wireshark	11
Wprowadzenie	11
Wygląd okna programu Wireshark	12
Menu główne	15
Poruszanie się po liście pakietów	16
Komunikacja w Wiresharku	17
Przechwytywanie ruchu ICMP	17
Obsługa projektów	23
Proste filtrowanie przechwyconych danych	30
Działanie komunikacji w DNS	30
Działanie mechanizmu three-way handshake	40
Profile w Wiresharku	44
Dodawanie nowych kolumn — kolumna różnicy czasu	45
Kolorowanie reguł	49
Działanie protokołu ARP	51
Konfiguracja funkcjonalności span-port na przełączniku	63
Komunikacja stacji roboczej w sieci z routerami	65
Użyteczne filtry	73
ROZDZIAŁ 2. Zaawansowane funkcjonalności programu Wireshark	75
Wprowadzenie	75
Ustawienia Wiresharka przed analizą	76
Podstawowe ustawienia	76
Co naprawdę oznacza Duplicate ACK?	79
Expert Info jako wykrywacz anomalii	82
Analiza dostarczania pakietów poza kolejnością	83
Symulacja ograniczenia przepustowości i analiza w Wiresharku	86
Przechwytywanie ruchu przez RSPAN i analizowanie sesji zdalnych	89
Analiza sesji TLS — co widać mimo szyfrowania?	93
Analiza sesji TLS — co widać po odszyfrowaniu ruchu?	103
Reprezentacja danych	112
Filtry wyświetlania	113
Ważne filtry do analizy ruchu sieciowego	113
Przykładowe i rzeczywiste scenariusze awarii i analiza z użyciem Wiresharka	116
Podsumowanie: jak naprawdę używać Wiresharka w praktyce?	118
Użyteczne filtry	120

ROZDZIAŁ 3. Przechwytywanie i analiza ruchu w warstwie drugiej OSI	122
Wprowadzenie	122
Protokoły warstwy drugiej modelu OSI	122
Protokół STP	122
Problem burzy ogłoszeniowej	126
Sieci VLAN	133
Ramki sterujące. Analiza protokołów CDP i LLDP	137
Protokół CDP	137
Protokół LLDP	140
Adresowanie MAC i analiza tablic MAC	142
Pozostałe ramki pojawiające się w sieciach komputerowych i ich analiza	146
Ramki LLC/SNAP	146
LACP/EtherChannel	148
Przydatne filtry warstwy drugiej	153
Użyteczne filtry	156
ROZDZIAŁ 4. Analiza włamań i bezpieczeństwo w warstwie drugiej OSI	158
Skanowanie sieci	158
Przeprowadzanie skanowania sieci za pomocą nmap	158
Skan TCP SYN	160
Skan UDP	162
Problemy z protokołem ARP (ARP poisoning i MiTM)	165
MAC flooding (przepełnienie tablicy CAM)	170
Wykrywanie STP manipulation	174
Użyteczne filtry	179
ROZDZIAŁ 5. Przechwytywanie i analiza ruchu w warstwie trzeciej OSI	181
Wstęp	181
Analiza pakietów IP	181
Analiza TTL i Hop Limit	182
Analiza polecenia traceroute	184
Wykrywanie duplikatów adresów IPv4	186
Wykrywanie duplikatów adresów IPv6	189
Warstwa trzecia widziana oczami TCP	191
Geolokalizacja w Wiresharku	209
GeoLite2 — darmowe dane geolokalizacyjne	209
Integracja z Wiresharkiem	210
Problemy z fragmentacją pakietów	212
Analiza routingu na przykładzie protokołu OSPF	222
Analiza problemów routingu na podstawie protokołu OSPF	227
Protokół DHCP w programie Wireshark	231
Użyteczne filtry	233
ROZDZIAŁ 6. Analiza włamań i bezpieczeństwo w warstwie trzeciej OSI	235
Wprowadzenie	235
Warstwa trzecia w analizie bezpieczeństwa — zakres i ograniczenia	235
IP spoofing. Podsywanie się pod adres źródłowy w warstwie trzeciej OSI	238
Nadużycie protokołu ICMP jako mechanizmu zakłócania komunikacji	241
ICMP reconnaissance	242
ICMP error abuse — trzy modele wymuszania reakcji systemu	246

Omijanie zabezpieczeń za pomocą fragmentacji danych	251
Fragmentacja jako technika zaciemniania i utrudniania analizy	252
Wygaśnięcie bufora rekonstrukcji (fragment timeout)	256
TTL manipulation i abnormal TTL — interpretacja	259
anomalii bezpieczeństwa	259
Tunelowanie IP w warstwie trzeciej: IP-in-IP i GRE	265
Manipulacje routinglem i analiza incydentów	
w protokołach routingu (OSPF)	278
Anomalie w zestawianiu sąsiedztwa OSPF. Analiza w Wiresharku	279
Studium incydentu laboratoryjnego	284
Anomalie związane z próbami przejścia hasel w OSPF	289
Użyteczne filtry	293
ROZDZIAŁ 7. Analiza ruchu w wyższych warstwach modelu OSI	294
Wprowadzenie	294
Analiza usług nazw — DNS	294
Standardowa komunikacja DNS	295
Opóźnienia i limity czasu (timeouts)	297
NXDOMAIN i błędne odpowiedzi	301
NODATA	302
Cache poisoning i ślady w ruchu	304
DNS jako źródło danych wywiadowczych	307
Fałszywy DHCP	311
Wzorce zachowań podczas procesu uwierzytelniania	316
Protokoły tekstowe i wyciek danych w postaci jawnej	326
Symulacja i diagnostyka problemów z TLS	334
Poczta elektroniczna jako wektor ataku. Analiza i detekcja	
w ruchu sieciowym	345
Analiza sesji pocztowych	345
Przykładowe ataki na pocztę elektroniczną i ich wykrywanie	350
Przechwytywanie i analiza protokołu QUIC	358
Pakiet Client Hello	366
Komunikat QUIC — Server Hello	368
Protokoły komunikacyjne	369
Funkcje protokołu UDP	370
Funkcje protokołu TCP	373
Analiza ruchu multimedialnego	381
Symulacja degradacji jakości VoIP w laboratorium domowym	400
Użyteczne filtry	413
ROZDZIAŁ 8. Funkcje statystyczne i analityczne programu Wireshark	415
Wprowadzenie	415
Podstawowa ocena pliku przechwytywania	415
Analiza różnego rodzaju ruchu	418
Analiza ruchu HTTP	418
Zaawansowana analiza wydajności i zachowania sieci	422
Analiza TCP na podstawie wykresów	436
Analiza warstwy aplikacyjnej pod kątem wydajności	450
Ocena wiarygodności przechwycenia oraz analiza strat i fragmentacji	465
Inne statystyki	470

Automatyzacja i rozszerzanie możliwości programu Wireshark —	
skrypty Lua	474
Użyteczne filtry	480
Najważniejsze pojęcia i skróty	481
Zakończenie	487
Bibliografia	488
Skorowidz	489

ROZDZIAŁ 4.

Analiza włamań i bezpieczeństwo w warstwie drugiej OSI

Skanowanie sieci

Zwykle każde włamanie do sieci komputerowej zaczyna się od rozpoznania terenu. W profesjonalnym języku nazywa się to rekonesansem sieci. Rekonesans jest pierwszym krokiem do włamania. Istnieje ktoś (bo za maszyną zawsze kryje się człowiek, przynajmniej na razie), kto nie będąc częścią Twojej sieci, nagle zaczyna próbować ją zrozumieć. Przeprowadza rekonesans.

Jeśli jest intruz, któremu udało się podłączyć do sieci, to w zasadzie ten czuje się ponieważ zdezorientowany. Dzieje się tak, ponieważ sieć (szczególnie w większych przedsiębiorstwach) jest bardzo dynamiczna. Mamy w niej wiele hostów, które potrafią pojawiać się i znikać nagle, tysiące zapytań różnego rodzaju i tyle samo odpowiedzi. Zmieniające się usługi i trudny do przewidzenia pozostały ruch nic potencjalnemu włamywaczowi nie mówią. Więc żeby mógł wykonać kolejny krok, musi zdobyć chociaż szczątkowe informacje o urządzeniach.

I tu dochodzimy do sedna, czyli technik skanowania i ich wykrywania podczas rutynowych audytów czy monitoringu sieci. Technik jest wiele i niektóre działają typowo w warstwie drugiej, inne badają porty, np. TCP czy UDP, jeszcze inne używają ICMP. Oczywiście wszystkie one mają wspólny mianownik, jakim jest poznanie sieci. Prawda jest taka, że każdy rekonesans zostawia pewne ślady, trzeba tylko być czujnym i umieć je rozpoznawać. W tym rozdziale postaram się nieco ten temat przybliżyć.

Pewne znaki w sieci mogą zwiastować próby jej skanowania, a nawet włamania. Strumienie ARP, bardzo wiele komunikatów SYN czy odpowiedzi ICMP mogą wzbudzić Twoje podejrzenia. Dlatego w dalszej części przeprowadzimy kilka analiz, ale na rzeczywistych przykładach. Będzie do tego potrzebny GNS3 (lub rzeczywisty sprzęt) oraz, co oczywiste, program Wireshark. Dzięki temu zobaczymy, jak z perspektywy analityka wygląda początek włamania i jak wcześnie można je dostrzec, zanim zmieni się w poważny incydent.

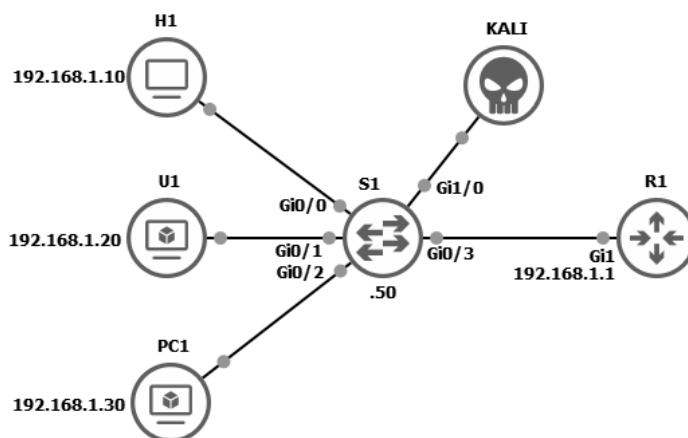
Przeprowadzanie skanowania sieci za pomocą nmap

No to zaczynamy zabawę. Jak już wspomniałem, nie chcę tylko pisać, co się pojawi, wklejać obrazki i tyle. Idea tej książki jest taka, że robię wszystko, abyś mógł powtórzyć testy u siebie w laboratorium.

W pierwszej kolejności budujemy niewielką sieć. Spójrz na rysunek 4.1. W sieci znajdują się trzy stacje robocze (Windows, Ubuntu i VPCS), przełącznik oraz router. Oczywiście jest też w sieci ten zły, czyli Kali Linux.

RYСУNEK 4.1.

Niewielka sieć ze stacjami roboczymi, przełącznikiem i routerem



Zacniemy od pierwszego rozpoznania wstępnego, czyli od narzędzia arp-scan. To proste narzędzie wysyła zapytania ARP na całą przestrzeń adresową. Można powiedzieć, że jest to technika bardzo prymitywna i będzie w tym trochę racji. Ale jest bardzo skuteczna. Dlaczego? Generalnie warstwa druga w sieciach z IPv4 działa bez żadnego mechanizmu, który ukrywałby adres MAC. Tak więc jeśli jakaś stacja zapyta, to cały segment usłyszy to pytanie, a najzwyczajniej w świecie właściciel tego adresu po prostu odpowie. Jak się za chwilę przekonasz, te zapytania w sieci widać, ale jeżeli w sieci jest naprawdę bardzo dużo ruchu, to mogą one po prostu „wpełznąć” niezauważone.

Przechodzimy do prostej symulacji, czegoś w rodzaju rozpoznania sieci. Na rysunku 4.2 znajduje się cały przeprowadzony skan sieci i otrzymany wynik. Użyto komendy `sudo arp-scan - localnet`. Jeśli atakowana sieć jest znana, to możesz zamiast parametru `- localnet` podać bezpośrednio adres sieci lub prefiks.

```
(kali@kali)-[~]
$ sudo arp-scan --localnet
Interface: eth0, type: EN10MB, MAC: 00:0c:29:8a:c8:d1, IPv4: 192.168.1.100
WARNING: Cannot open MAC/Vendor file ieee-oui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file mac-vendor.txt: Permission denied
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.1.1      0c:7a:f1:08:00:00      (Unknown)
192.168.1.1      0c:7a:f1:08:00:00      (Unknown) (DUP: 2)
192.168.1.10     12:34:12:34:12:34      (Unknown: locally administered)
192.168.1.20     00:0c:29:96:82:48      (Unknown)
192.168.1.30     00:50:79:66:68:00      (Unknown)
192.168.1.30     00:50:79:66:68:00      (Unknown) (DUP: 2)
192.168.1.50     0c:49:37:a3:80:01      (Unknown)
192.168.1.50     0c:49:37:a3:80:01      (Unknown) (DUP: 2)
192.168.1.50     0c:49:37:a3:80:01      (Unknown) (DUP: 3)

9 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 1.855 seconds (138.01 hosts/sec). 5 responded
(kali@kali)-[~]
$
```

RYСУNEK 4.2. Wynik przeprowadzonego skanowania sieci

Rozpoczęcie skanowania od razu wywołało lawinę zapytań ARP. Na rysunku 4.3 widać wiele zapytań ARP, ale co ciekawe, nie mija nawet sekunda od uruchomienia skanowania. Zauważ, że każde takie zapytanie trafia pod adres rozgłoszeniowy ff:ff:ff:ff:ff:ff. Tak więc przełącznik przekazał zapytania na swoje wszystkie aktywne interfejsy należące do tego samego VLAN-u. Intruz tym zapytaniem poniekąd zmusił całą sieć do odsłonięcia części struktury. Odpowiedziały wszystkie aktywne stacje.

No.	Time	DELTA	Source	Destination	Protocol	Length	Info
28	12.899797	0.881223	0c:49:37:a3:00:00	01:80:c2:00:00:00	STP	60	Conf. Root = 32768/1/0c:49:37:a3:00:00 Cost = 0 Port = 0x8001
29	13.020261	0.120464	192.168.154.129	192.168.154.1	TCP	141	80 → 63390 [PSH, ACK] Seq=1132 Ack=1 Win=501 Len=87
30	14.021897	1.001636	192.168.154.129	192.168.154.1	TCP	141	80 → 63390 [PSH, ACK] Seq=1219 Ack=1 Win=501 Len=87
31	14.907454	0.885557	0c:49:37:a3:00:00	01:80:c2:00:00:00	STP	60	Conf. Root = 32768/1/0c:49:37:a3:00:00 Cost = 0 Port = 0x8001
32	15.024040	0.116586	192.168.154.129	192.168.154.1	TCP	141	80 → 63390 [PSH, ACK] Seq=1306 Ack=1 Win=501 Len=87
33	15.873399	0.849359	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.0? Tell 192.168.1.100
34	15.875629	0.002330	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.1? Tell 192.168.1.100
35	15.878218	0.002589	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.2? Tell 192.168.1.100
36	15.878964	0.000746	0c:7a:f1:08:00:00	0c:29:8a:c8:d1	ARP	60	192.168.1.1 is at 0c:7a:f1:08:00:00
37	15.879384	0.000420	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.3? Tell 192.168.1.100
38	15.880480	0.001096	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.4? Tell 192.168.1.100
39	15.882109	0.001629	0c:7a:f1:08:00:00	0c:29:8a:c8:d1	ARP	60	192.168.1.1 is at 0c:7a:f1:08:00:00
40	15.882560	0.000451	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.5? Tell 192.168.1.100
41	15.885616	0.003056	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.6? Tell 192.168.1.100
42	15.888356	0.002740	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.7? Tell 192.168.1.100
43	15.890424	0.002068	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.8? Tell 192.168.1.100
44	15.891042	0.000618	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.9? Tell 192.168.1.100
45	15.892166	0.001124	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.10? Tell 192.168.1.100
46	15.892206	0.000040	12:34:12:34:12:34	00:0c:29:8a:c8:d1	ARP	42	192.168.1.10 is at 12:34:12:34:12:34
47	15.894666	0.002460	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.11? Tell 192.168.1.100
48	15.897880	0.003214	00:0c:29:8a:c8:d1	ff:ff:ff:ff:ff:ff	ARP	60	Who has 192.168.1.12? Tell 192.168.1.100

RYСУNEK 4.3. Przechwycone ramki ARP

Na rysunku widać zapisy:

```
192.168.1.1 is at 0c:7a:f1:08:00:00
192.168.1.10 is at 12:34:12:34:12:34
```

Czyli: wystąpił w sumie prosty atak i intruz ma już listę aktywnych urządzeń i ich adresy IP oraz adresy MAC. Dzięki temu wie, jaki to sprzęt i producent. To cenna wiedza jak na początek.

Na co warto zwrócić uwagę podczas analizy? Dla tego typu rekonesansu charakterystyczne jest to, że ten rodzaj skanowania tworzy bardzo wyraźny wzorec czasowy. Zatem od razu zobaczysz serię ramek z tym samym adresem MAC źródłowym w bardzo krótkich odstępach czasu. Mało tego; zwykle będzie to *ARP Request* skierowany pod adres MAC rozgłoszeniowy. Dla analityka charakterystyczna jest seria ramek *ARP Request* wysyłanych z jednego adresu MAC w bardzo krótkich odstępach czasu, zwykle pod adres rozgłoszeniowy ff:ff:ff:ff:ff:ff. Taki wzorec często wskazuje na rekonesans lub automatyczne sondowanie sieci, choć sam w sobie nie stanowi jeszcze niepodważalnego dowodu ataku. Ostateczna ocena powinna uwzględniać kontekst sieci, profil normalnego ruchu oraz zachowanie pozostałych hostów.

Wiedza ta jest kluczowa w praktyce śledczej. Jeśli w logach przełącznika lub przechwyconym ruchu sieciowym występuje rytmiczna sekwencja protokołu ARP w ciągu kilku sekund, to można z dużym prawdopodobieństwem przyjąć, że w tym czasie trwało aktywne rozpoznanie lub automatyczne sondowanie sieci.

Skan TCP SYN

Intruz w poprzednim kroku poznał, z jaką siecią ma do czynienia. Zna już adresy IP, ale jeszcze nie wie, jaka usługa jest uruchomiona na jakich urządzeniach. Teraz musi więc sprawdzić, jakie usługi są otwarte, aby wiedzieć, które można zaatakować i na jakich urządzeniach.

Chciałbym na tym etapie wspomnieć, że użyte komendy celowo skanują całą sieć, ale musisz mieć świadomość, że intruz może wykonać skan tylko jednego konkretnego urządzenia oraz usługi. Wtedy nie zobaczysz aż tylu wiadomości. Natomiast chodzi tutaj o pokazanie Ci pewnego wzorca działania i rodzaju komunikatów, jakie się pojawiają w takim przypadku.

Najbardziej typowym i najczęściej używanym skanem w analizie sieci jest *SYN scan*. *SYN scan* uruchamiany opcją `-sS` jest szybki i relatywnie mało inwazyjny, ponieważ nie dochodzi w nim do pełnego ustanowienia połączenia TCP. Z tego powodu część usług aplikacyjnych może nie odnotować takiego kontaktu jako pełnego połączenia, jednak ruch ten nadal może być rejestrowany przez zapory, systemy IDS/IPS, mechanizmy audytu systemowego lub inne narzędzia monitorujące.

Spójrz na rysunek 4.4, na którym widać użycie polecenia `sudo nmap -sS 192.168.1.0/24` oraz fragment wyniku. Wydane polecenie oznacza, że dla każdego hosta w podsieci zostaną wysłane pakiety SYN na wybrane porty TCP, `nmap` sprawdzi wynik i wyświetli odpowiednie informacje.

RYСУNEK 4.4.

Wynik skanowania całej sieci

```
(kali@kali) [ ~ ]
$ sudo nmap -sS 192.168.1.0/24
sudo: password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-11 14:56 EST
Nmap scan report for 192.168.1.1
Host is up (0.18s latency).
Not shown: 996 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
23/tcp    open  telnet
80/tcp    open  http
443/tcp   open  https
MAC Address: 0C:7A:F1:08:00:00 (Unknown)

Nmap scan report for 192.168.1.10
Host is up (0.00017s latency).
Not shown: 992 closed tcp ports (reset)
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3300/tcp  open  ceph
3389/tcp  open  ms-wbt-server
5000/tcp  open  upnp
8088/tcp  open  radan-http
MAC Address: 12:34:12:34:12:34 (Unknown)

Nmap scan report for 192.168.1.20
Host is up (0.00010s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE
```

A teraz spójrz na rysunek 4.5, na którym znajdziesz prawdziwe zamieszanie. Tylko na pierwszy rzut oka wydaje się to nie mieć sensu. Kiedy sam przeprowadzisz taki atak, zauważysz, że w Wiresharku pojawia się bardzo charakterystyczny wzorec ramek. Schemat dla portu zamkniętego wygląda zwykle następująco: skaner wysyła SYN, a host odpowiada RST/ACK. Dla portu otwartego odpowiedź hosta ma postać SYN/ACK, po czym skaner zwykle odsyła RST, aby nie kończyć pełnego połączenia. To właśnie powtarzalność tych sekwencji pozwala analitykowi rozpoznać półotwarty skan SYN.

No.	Time	DELTA	Source	Destination	Protocol	Length	Info	Time to Live	Identification
21299	70.661218	0.002141	192.168.1.50	192.168.1.100	TCP	60	5950 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x0eaa
21300	70.669294	0.008076	192.168.1.100	192.168.1.50	TCP	60	60456 → 7920 [SYN] Seq=0 Win=1024 Len=0 MSS=1460	44	0xe3dc
21301	70.671555	0.002261	192.168.1.50	192.168.1.100	TCP	60	7920 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0xc989
21302	70.673843	0.002288	192.168.1.50	192.168.1.100	TCP	60	7920 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0xa022
21303	70.679446	0.005681	192.168.1.100	192.168.1.50	TCP	60	60456 → 49156 [SYN] Seq=0 Win=1024 Len=0 MSS=1460	47	0xe0ab
21304	70.681745	0.002299	192.168.1.50	192.168.1.100	TCP	60	49156 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x2906
21305	70.683918	0.002192	192.168.1.50	192.168.1.100	TCP	60	49156 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x0950
21306	70.689507	0.005629	192.168.1.100	192.168.1.50	TCP	60	60456 → 5200 [SYN] Seq=0 Win=1024 Len=0 MSS=1460	56	0xd3d3
21307	70.691759	0.002192	192.168.1.50	192.168.1.100	TCP	60	5200 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0xd5ae
21308	70.694040	0.002281	192.168.1.50	192.168.1.100	TCP	60	5200 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x7672
21309	70.699800	0.005760	192.168.1.100	192.168.1.50	TCP	60	60456 → 8651 [SYN] Seq=0 Win=1024 Len=0 MSS=1460	39	0x035c
21310	70.701924	0.002124	192.168.1.50	192.168.1.100	TCP	60	8651 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x52d9
21311	70.704101	0.002177	192.168.1.50	192.168.1.100	TCP	60	8651 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x568d
21312	70.710099	0.005998	192.168.1.100	192.168.1.50	TCP	60	60456 → 7920 [SYN] Seq=0 Win=1024 Len=0 MSS=1460	41	0x0ce5
21313	70.712311	0.002212	192.168.1.50	192.168.1.100	TCP	60	7920 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x1143
21314	70.714563	0.002252	192.168.1.50	192.168.1.100	TCP	60	7920 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0xad9b
21315	70.720424	0.005861	192.168.1.100	192.168.1.50	TCP	60	60456 → 990 [SYN] Seq=0 Win=1024 Len=0 MSS=1460	49	0xd5ab
21316	70.722575	0.002151	192.168.1.50	192.168.1.100	TCP	60	990 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x407c
21317	70.724850	0.002275	192.168.1.50	192.168.1.100	TCP	60	990 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x0e8a
21318	70.730820	0.005970	192.168.1.100	192.168.1.50	TCP	60	60456 → 61532 [SYN] Seq=0 Win=1024 Len=0 MSS=1460	40	0xad02
21319	70.733006	0.002180	192.168.1.50	192.168.1.100	TCP	60	61532 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x021a
21320	70.736391	0.002323	192.168.1.50	192.168.1.100	TCP	60	61532 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x7b50
21321	70.741500	0.005238	192.168.1.100	192.168.1.50	TCP	60	60456 → 51103 [SYN] Seq=0 Win=1024 Len=0 MSS=1460	50	0x5b77
21322	70.743727	0.002158	192.168.1.50	192.168.1.100	TCP	60	51103 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x3033
21323	70.745998	0.002271	192.168.1.50	192.168.1.100	TCP	60	51103 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x5090
21324	70.752055	0.006087	192.168.1.100	192.168.1.50	TCP	60	60456 → 3371 [SYN] Seq=0 Win=1024 Len=0 MSS=1460	37	0x1a75
21325	70.754262	0.002207	192.168.1.50	192.168.1.100	TCP	60	3371 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0xcce8
21326	70.756402	0.002200	192.168.1.50	192.168.1.100	TCP	60	3371 → 60456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	255	0x5e66

RYSUNEK 4.5. Przechwycony ruch TCP

Rysunek obrazuje ten schemat:

```

192.168.1.50 → 192.168.1.100 [SYN]
192.168.1.100 → 192.168.1.50 [RST, ACK]
192.168.1.50 → 192.168.1.100 [SYN]
192.168.1.100 → 192.168.1.50 [RST, ACK]

```

Zauważ, że wynik jest niemalże geometryczny, powtarzalny i trudny do pomylenia z naturalnym ruchem sieci.

Dla analityka *SYN scan* to jeden z najbardziej jednoznacznych sygnałów, że w sieci dzieje się coś niepokojącego. Układ opcji TCP, wartości okna i charakter sekwencji pakietów mogą być pomocne w rozpoznaniu narzędzia, jednak nie stanowią jednoznacznego dowodu, ponieważ zależą od wersji nmap, systemu operacyjnego, typu skanu oraz urządzeń pośredniczących. Od razu rzuca się w oczy wysoka częstotliwość ramek. Zauważ, że kolejne komunikaty SYN wysyłane są co 1 – 3 ms. Charakterystyczny jest również bardzo powtarzalny wzorzec, przedstawiający identyczny host źródłowy, różne porty docelowe, następnie brak dokończenia handshake'u, bo host odpowiada SYN/ACK, ale intruz nie finalizuje handshake'u. Warto również zwrócić uwagę, jeśli w sieci pojawia się ruch do portów, które normalnie nigdy nie są używane.

W wielu scenariuszach nmap wykazuje powtarzalny wzorzec kolejności odpytywania portów, ale nie należy zakładać, że zawsze skanuje je liniowo ani że sama kolejność pozwala pewnie odtworzyć intencję atakującego. Pomaga to jednak przewidzieć jego kolejny krok. Tego typu cechy bywają pomocne w rozpoznaniu narzędzia lub systemu źródłowego, ale nie należy traktować ich jako jednoznacznego dowodu, gdyż podobny układ może zależeć również od stosu TCP/IP, wersji narzędzia i urządzeń pośredniczących.

Skan UDP

Skanowanie UDP jest nieco inne. Dzieje się tak dlatego, że protokół UDP nie daje atakującemu wygodnych sygnałów, tak jak to było w TCP. W TCP pary SYN–RST lub SYN–SYN/ACK formują czytelny język stanu portu. W UDP reakcje są bardziej rozproszone, a większość sensu bierze się nie z samego datagramu, ale z tego, co system operacyjny robi w odpowiedzi na jego brak.

Tym razem do przeprowadzenia skanu UDP użyto w KALI polecenia `nmap -sU 192.168.1.0/24`. Spójrz na rysunek 4.6. Widać na nim wydane polecenie oraz fragment wyniku. Zatrzymajmy się na chwilę przy tym wyniku.

RYSUNEK 4.6.

Wynik skanowania
UDP

```
(kali㉿kali)-[~]
└─$ sudo nmap -sU 192.168.1.0/24
[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-11 15:50 EST
Nmap scan report for 192.168.1.1
Host is up (0.0062s latency).
All 1000 scanned ports on 192.168.1.1 are in ignored states.
Not shown: 1000 closed udp ports (port-unreach)
MAC Address: 0C:7A:F1:08:00:00 (Unknown)

Nmap scan report for 192.168.1.10
Host is up (0.00026s latency).
Not shown: 989 closed udp ports (port-unreach)
PORT      STATE      SERVICE
137/udp    open       netbios-ns
138/udp    open|filtered netbios-dgm
162/udp    open|filtered snmptrap
500/udp    open|filtered isakmp
514/udp    open|filtered syslog
1900/udp   open|filtered upnp
3389/udp   open|filtered ms-wbt-server
4500/udp   open|filtered nat-t-ike
5050/udp   open|filtered mmcc
5353/udp   open|filtered zeroconf
5355/udp   open|filtered llmnr
MAC Address: 12:34:12:34:12:34 (Unknown)

Nmap scan report for 192.168.1.20
Host is up (0.00034s latency).
Not shown: 998 closed udp ports (port-unreach)
PORT      STATE      SERVICE
631/udp    open|filtered ipp
5353/udp   open|filtered zeroconf
MAC Address: 00:0C:29:96:82:48 (VMware)

Nmap scan report for 192.168.1.30
Host is up (0.069s latency).
```

Zwróć uwagę na to, że wynik skanowania UDP ma zupełnie inny charakter niż skan TCP. W raporcie od razu rzuca się w oczy, że router 192.168.1.1 nie zwrócił żadnych widocznych otwartych portów UDP. Zatem otrzymujesz informację, że wszystkie porty znajdują się w stanie ignored states. Jeżeli skanowany port UDP jest zamknięty, system może odesłać *ICMP Destination unreachable/Port unreachable*, co pozwala nmap oznaczyć port jako closed. W praktyce takie odpowiedzi mogą jednak podlegać filtrowaniu lub ograniczaniu szybkości, zwłaszcza na urządzeniach sieciowych, dlatego brak odpowiedzi nie musi oznaczać portu otwartego i często skutkuje stanem open/filtered. Dla atakującego oznacza to, że router R1 nie jest atrakcyjnym celem na tym etapie. Dla analityka natomiast jest to sygnał, że router zachowywał się przewidywalnie, bo odpowiedzi ICMP spełniają swoje zadanie.

Dalej na rysunku znajdują się wyniki dla hostów. Dostyc szczegółowo o otwartych portach informuje 192.168.1.10. Widać tam cały szereg portów oznaczonych jako open/filtered. Można powiedzieć, że jest to stan, który wprowadza pewien rodzaj niejednoznaczności. Dlaczego tak jest? W zasadzie nmap nie wie, czy port faktycznie jest otwarty, czy ruch został odfiltrowany. Dzieje się tak dlatego, że w UDP brakuje odpowiednika tego, co świetnie działa w TCP, czyli po prostu odpowiedzi.

Jak już wiesz, protokół UDP ma bardzo prostą naturę. Urządzenie wysyła pakiet i nie dostaje potwierdzenia, czy dotarł, czy nie. Milczenie nie ma w UDP żadnego specjalnego znaczenia. Ale w skanowaniu nmap to milczenie jest źródłem całego problemu. Kiedy program nmap wysyła pakiet UDP na jakiś port, mogą wydarzyć się trzy rzeczy. Pierwsza możliwość: host odpowiada *ICMP Port unreachable*. W takim wypadku nmap wie, że port jest zamknięty. Druga możliwość: host odpowie jakąś prawdziwą odpowiedzią UDP. W takim wypadku nmap wie, że port jest otwarty. I trzecia możliwość: host nie odpowie. Tutaj właśnie pojawia się problem. To milczenie można interpretować na dwa sposoby. Pierwsza interpretacja jest taka, że port faktycznie jest otwarty i usługa przyjmuje pakiet mimo braku ICMP. Druga interpretacja to, że firewall po drodze odrzuca pakiety bez słowa, celowo myląc intruza. nmap nie ma jak tego rozróżnić. Dlatego oznacza te porty jako *open/fil tered*.

Po zakończonym ataku przyjrzymy się temu, co udało się przechwycić w trakcie jego trwania. Rysunek 4.7 przedstawia przechwycony ruch.

No.	Time	DELTA	Source	Destination	Protocol	Length	Info
3961	60.668481	0.001022	192.168.1.100	192.168.1.20	UDP	42	36871 → 113 Len=0
3962	60.668731	0.000250	192.168.1.100	192.168.1.20	UDP	42	36871 → 113 Len=0
3963	60.763683	0.094952	0c:49:37:a3:00...01:80:c2:00:00...		STP	60	Conf. Root = 32768/1/0c:49:37:a3:00:00 Cost = 0 P
3964	60.767504	0.003821	192.168.1.100	192.168.1.1	UDP	82	36871 → 40019 Len=40
3965	60.768701	0.001197	192.168.1.1	192.168.1.100	ICMP	70	Destination unreachable (Port unreachable)
3966	60.979669	0.210968	192.168.1.100	192.168.1.50	UDP	82	36873 → 40019 Len=40
3967	60.981966	0.002297	192.168.1.50	192.168.1.100	ICMP	70	Destination unreachable (Port unreachable)
3968	61.068153	0.086187	192.168.1.100	192.168.1.20	UDP	60	36873 → 113 Len=0
3969	61.068272	0.000119	192.168.1.20	192.168.1.100	ICMP	70	Destination unreachable (Port unreachable)
3970	61.069969	0.001697	192.168.1.100	192.168.1.20	UDP	42	36873 → 113 Len=0
3971	61.071095	0.001126	192.168.1.20	192.168.1.100	ICMP	70	Destination unreachable (Port unreachable)
3972	61.072777	0.001682	192.168.1.50	192.168.1.100	ICMP	70	Redirect (Redirect for host)
3973	61.074131	0.001354	192.168.1.100	192.168.1.20	UDP	60	36873 → 113 Len=0
3974	61.075286	0.001075	192.168.1.100	192.168.1.10	UDP	60	36873 → 20380 Len=0
3975	61.075324	0.000110	192.168.1.10	192.168.1.100	ICMP	70	Destination unreachable (Port unreachable)
3976	61.076194	0.000870	192.168.1.100	192.168.1.10	UDP	42	36873 → 20380 Len=0
3977	61.076479	0.000285	192.168.1.10	192.168.1.100	ICMP	70	Destination unreachable (Port unreachable)
3978	61.077991	0.001512	192.168.1.10	192.168.1.100	ICMP	70	Destination unreachable (Port unreachable)
3979	61.090089	0.012098	192.168.154.129	192.168.154.1	TCP	141	80 → 49777 [PSH, ACK] Seq=5312 Ack=1 Win=501 Len=87
3980	61.300109	0.290020	192.168.1.100	192.168.1.50	UDP	82	36871 → 34038 Len=40
3981	61.568187	0.188078	192.168.1.100	192.168.1.1	UDP	82	36871 → 34038 Len=40
3982	61.570421	0.002234	192.168.1.1	192.168.1.100	ICMP	70	Destination unreachable (Port unreachable)
3983	61.781161	0.210740	192.168.1.100	192.168.1.50	UDP	82	36873 → 34038 Len=40
3984	61.783170	0.002009	192.168.1.50	192.168.1.100	ICMP	70	Destination unreachable (Port unreachable)
3985	61.868250	0.085080	192.168.1.100	192.168.1.20	UDP	82	36871 → 50612 Len=40
3986	61.868930	0.000680	192.168.1.100	192.168.1.20	UDP	82	36871 → 50612 Len=40
3987	61.869196	0.000266	192.168.1.100	192.168.1.20	UDP	82	36871 → 50612 Len=40

RYСУNEK 4.7. Pakiety UDP przechwycone w trakcie skanowania

Zauważ, że nmap wysyła krótkie datagramy UDP do kolejnych portów i czeka na reakcję. Te datagramy są oszczędne, pozbawione treści, bo ich celem nie jest nawiązanie połączenia, lecz wywołanie odpowiedzi systemu operacyjnego po drugiej stronie. W Wiresharku ruch ten od razu wyróżnia się na tle wszystkiego, co znamy z normalnego użytkownika sieci. Jest to bardzo intensywne i równomierne odpytywanie każdego portu. W krótkich, milisekundowych odstępach pojawiają się datagramy UDP wysyłane do hostów w sieci: 192.168.1.10, 1.20, 1.30, a także do routera 192.168.1.1. Jest to bardzo podejrzany ruch. Dalej mamy odpowiedzi *ICMP Destination unreachable*, które poszczególne systemy wysyłają, gdy dany port UDP jest zamknięty.

Dla analityka odpowiedzi ICMP mogą stanowić cenną wskazówkę interpretacyjną, ponieważ ich typ, kod, częstotliwość i sposób generowania pozwalają wnioskować o zachowaniu hosta lub urządzenia pośredniczącego. Nie należy jednak traktować tych cech jako samodzielnego, jednoznacznego identyfikatora systemu operacyjnego lub roli urządzenia.

Zauważ, że router odpowiada inaczej niż hosty końcowe; widzimy nawet moment, w którym router udziela komunikatu `Redirect`, pokazując, że intruz chwilowo próbował wysłać pakiet przez bramę, mimo że cel znajdował się w tej samej sieci.

To, co najważniejsze dla analityka, to skala i intensywność. Normalna sieć w zasadzie nie generuje strumieni *ICMP Port unreachable* w takich ilościach. Mogą się zdarzyć pojedyncze komunikaty, gdy aplikacja próbuje kontaktować się z usługą, której nie ma, ale nie zobaczymy ciągłej serii odpowiedzi z kilku hostów naraz, z równym, niemal matematycznym rytmem. Tutaj ICMP jest regularny, powtarzalny i bez dłuższych przerw. Jest to typowy znak sugerujący odpytywanie.

Problemy z protokołem ARP (ARP poisoning i MiTM)

W rozdziale 1. omówiłem protokół ARP. Wbrew pozorom protokół ten wciąż pełni niezwykle istotną funkcję w komunikacji. Szczególnie w komunikacji dotyczącej protokołu IPv4, który ciągle jest najczęściej używanym protokołem warstwy sieci. Dlatego zagrożenia, które dotyczą protokołu ARP, są nadal aktualne i mogą być niebezpieczne, jeśli na czas nie zostaną wykryte. Zobaczmy, w jaki sposób możesz wykryć anomalie związane z ARP w programie Wireshark. ARP to mechanizm mapowania adresów IP na adresy łącza (MAC), a jego podstawy opisuje RFC 826.

Atak polega na tym, że osoba niepowołana uruchamia program, który służy do oszukiwania kart sieciowych. Skutkiem działania takiego programu jest to, że stacja robocza, przesyłając zapytanie ARP, otrzymuje w odpowiedzi od atakującego adres MAC jego stacji roboczej (atakującej). Uaktualnia na tej podstawie swoją tablicę ARP. Dzieje się tak ze względu na to, że atakujący zwykle wysyła spreparowane *ARP_REPLY* (*opcode 2*) albo ogłoszenia (*gratuitous ARP*), które powodują nadpisanie wpisów ARP ofiary. Dzięki temu część ruchu ofiary, w szczególności ruch kierowany do podszywanego adresu IP, może zostać skierowana do hosta atakującego. Aby uzyskać pełny scenariusz MiTM, atakujący musi jeszcze zapewnić dalsze przekazywanie pakietów, w przeciwnym razie skutkiem może być jedynie zakłócenie komunikacji.

Pełny atak spoofingowy polega na tym, że atakujący tak przystosowuje sprzęt, że ofiara nie ma nawet świadomości, że jest podsłuchiwana. Wszystkie usługi uruchomione przez ofiarę działają tak, jak gdyby nic niepokojącego się nie działo.

W praktyce należy odróżnić *ARP Probe* od *gratuitous ARP*, czyli *ARP Announcement*. *ARP Probe* służy do sprawdzenia, czy planowany adres IPv4 nie jest już używany w sieci, natomiast *gratuitous ARP* jest rozgłoszeniowym ogłoszeniem własnego mapowania IP/MAC, stosowanym m.in. do odświeżania tablic ARP innych hostów lub do obsługi scenariuszy przełączenia awaryjnego. Samo wystąpienie *gratuitous ARP* nie jest z definicji złośliwe, lecz w analizie bezpieczeństwa należy zwracać uwagę na jego częstotliwość, kontekst i spójność z topologią sieci.

Nie każde częste lub nietypowe występowanie zapytań ARP oznacza atak. W wielu sieciach można obserwować wzmożony ruch ARP wynikający z normalnej pracy urządzeń, rekonfiguracji hostów, odświeżania wpisów cache'u, pracy środowisk wirtualnych, mechanizmów wysokiej dostępności albo procesów wykrywania konfliktu adresów. Z drugiej

strony ruch ARP, który nie pasuje do normalnego profilu sieci, powinien zwrócić uwagę analityka. Dlatego ocena nie powinna opierać się na pojedynczym pakiecie ani jednej liczbie, lecz na kontekście, częstotliwości, powtarzalności i spójności mapowań IP/MAC.

Nie istnieje uniwersalny próg liczby pakietów ARP, po przekroczeniu którego można automatycznie stwierdzić atak. Ocena powinna się opierać na profilu normalnego ruchu w danej sieci, typie urządzeń, ich funkcjach oraz obserwowanych wzorcach czasowych. Pamiętaj, że w ocenianiu takich zjawisk musisz być ostrożny. Nawet jeśli masz więcej komunikacji ARP, niż to zaznaczyłem, np. 5, a nawet 10 na sekundę, też nie musi to stanowić problemu. Zawsze warto szukać w takim przypadku podejrzanych wzorców. Sama liczba ARP w sieci zależy od tego, co faktycznie robią urządzenia w sieci. Jeśli wszystkie urządzenia łączą się tylko z jednym urządzeniem (np. z routerem, który łączy je dalej z siecią globalną), to zobaczysz niewielką liczbę ARP. Jeśli jednak urządzenia wysyłają np. okresowe wiadomości do wszystkich swoich sąsiadów, zobaczysz bardzo dużo komunikacji ARP. Pamiętaj jednak, że nie ma standardowego, uniwersalnego progu liczby pakietów ARP, który definiowałby atak. Najlepsza praktyka to zbudowanie typowego profilu ruchu ARP dla danej sieci i wykrywanie anomalii. Nie ma w żadnym RFC dokładnych bezpiecznych wartości, jednak na podstawie swoich doświadczeń podałem je powyżej, abyś miał jakieś rozeznanie.

Nie chcę tylko teoretyzować, dlatego najpierw chciałbym pokazać Ci, jak szybko przeprowadzić eksperyment, w którym możesz wywołać taki atak, a następnie już w domowych warunkach analizować jego skutki. Pamiętaj, by eksperyment przeprowadzić w sieci odseparowanej od rzeczywistej, najlepiej na maszynach wirtualnych z interfejsami ustawionymi na wirtualne VMnet, jeśli używasz np. VMware.

Na jednej maszynie wirtualnej uruchom system KALI, a na drugiej np. stację z systemem Linux lub Windows. Chodzi o to, aby połączyć te dwie maszyny poprzez interfejs wirtualny. Więcej nie będzie Ci potrzebne. Oczywiście sprawdź wcześniej, czy maszyny się ze sobą komunikują.



W ramach bardziej rozszerzonych ćwiczeń i do pełnej demonstracji ataku ARP spoofing/MiTM najlepiej przygotować co najmniej trzy węzły, tj. host atakujący, host ofiary oraz bramę lub trzeci host, przez który będzie przechodził ruch. Dwie maszyny wystarczają do obserwacji wybranych zjawisk ARP, ale nie oddają w pełni scenariusza klasycznego pośredniczenia w komunikacji.

Atak przeprowadzisz z maszyny KALI, wykorzystując program bettercap. Może się okazać, że program nie jest dostępny, dlatego wcześniej sprawdź to, wpisując w terminalu komendę `sudo bettercap -v`. Jeśli otrzymasz błąd, wówczas pobierz i zainstaluj odpowiedni pakiet, wydając najpierw komendę `sudo apt update`, a potem `sudo apt install bettercap -y`. Oczywiście w tym czasie nie zapomnij o podłączeniu maszyny wirtualnej do sieci Internet.

Po ewentualnej instalacji możesz przejść do sprawdzenia, jaki interfejs jest odpowiedni do komunikacji z drugą maszyną. Użyj komendy `ip addr show` i zapisz symbol interfejsu. Dalej przejdź komendą `sudo bettercap -iface [interfejs]` do trybu wydawania dodatkowych poleceń programu bettercap.

Najpierw wydaj komendę `net.probe on`, która może wykryć aktywne hosty widoczne w danym segmencie sieci. Następnie wpisz: `set arp.spoof.targets [adres IP stacji lub całej sieci]`. Kolejna komenda jest odpowiedzialna za uruchomienie ataku lokalnie: `set arp.spoof.internal true`. Aby uruchomić ataki, wydaj komendy `arp.spoof on` i `net.sniff on`.

W celu ich przerwania jako parametr wpisz `off`. Rysunek 4.8 przedstawia wydanie opisanych komend.

```
(kali@kali)-[~]
└─$ sudo bettercap -iface eth1
bettercap v2.33.0 (built for linux amd64 with go1.22.6) [type 'help' for a list of commands]

192.168.154.0/24 > 192.168.154.131  » [03:08:26] [sys.log] [ ] Could not find mac for
192.168.154.0/24 > 192.168.154.131  » net.probe on
[03:08:37] [sys.log] [inf] net.probe starting net.recon as a requirement for net.probe
192.168.154.0/24 > 192.168.154.131  » [03:08:37] [sys.log] [ ] net.probe probing 256 addresses on 192.168.154.0/24
192.168.154.0/24 > 192.168.154.131  » [03:08:37] [endpoint.new] endpoint fe80::2e3b:b235:4f42:366d detected as 00:50:56:c000:01 (VMware, Inc.).
192.168.154.0/24 > 192.168.154.131  » [03:08:39] [endpoint.new] endpoint 192.168.154.132 (DESKTOP-LS0S0M0) detected as 12:34:32:34:32:34.
192.168.154.0/24 > 192.168.154.131  » [03:08:41] [endpoint.new] endpoint 192.168.154.254 detected as 00:50:56:c2:00:04 (VMware, Inc.).
192.168.154.0/24 > 192.168.154.131  » set arp.spoof.targets 192.168.154.132
192.168.154.0/24 > 192.168.154.131  » set arp.spoof.internal true
192.168.154.0/24 > 192.168.154.131  » arp.spoof on
192.168.154.0/24 > 192.168.154.131  » [03:09:21] [sys.log] [ ] arp.spoof arp spoofer started targeting 254 possible network neighbours of 1 targets.
192.168.154.0/24 > 192.168.154.131  » net.sniff on
192.168.154.0/24 > 192.168.154.131  » net.sniff off
192.168.154.0/24 > 192.168.154.131  » arp.spoof off
```

RYСУNEK 4.8. Atak ARP spoof

Teraz, kiedy wiesz już, jak przeprowadzić atak, możesz zebrać dowolną ilość danych w programie Wireshark. W tym celu w trakcie ataku uruchom program Wireshark np. na stacji atakowanej lub na łączu wirtualnym.

Jak rozpoznać tego typu ataki w sieci?

Duża liczba *ARP Request* może wskazywać na etap rozpoznania lub aktywne sondowanie sieci przez narzędzie atakującego, jednak nie jest jeszcze istotą ataku ARP spoofing. Właściwe zatrucie cache'u ARP rozpoznaje się przede wszystkim po fałszywych *ARP Reply* albo powtarzających się *gratuitous ARP*, w których jeden adres MAC zostaje ogłoszony jako właściciel wielu adresów IP. Masowe *ARP Request* są zwykle objawem etapu sondowania lub działania funkcji wykrywania hostów, natomiast właściwe zatrucie cache'u ARP realizuje się przede wszystkim przez spreparowane *ARP Reply* albo powtarzające się ogłoszenia mapowania IP/MAC.

W pierwszej kolejności użyj filtru `arp`. Na rysunku 4.9 możesz zauważyć wiele ramek ARP. Są to masowo wysyłane pakiety *ARP Request*. Świadczą o tym zapisy takie jak np. `Who has 192.168.154.xxx? Tell 192.168.154.131`. Źródłem tych pakietów jest stacja z tym samym adresem MAC. Pakiety kierowane są pod adres MAC rozgłoszeniowy `ff:ff:ff:ff:ff:ff`. Na razie w sumie nic odkrywczego nie piszę, ale jest pewien szczegół, który powinien Cię zawsze zainteresować. Tak naprawdę informacja `Tell 192.168.154.131` oznacza, że jest to źródłowy adres, do którego każda ze stacji ma odpowiedzieć. I gdyby nie to, że w zaledwie kilkanaście sekund mamy już na liście prawie 20 000 takich pakietów, można by uznać to za poprawną komunikację. Ale liczba tych zapytań jest bardzo duża i w zasadzie mamy do czynienia z zalewaniem. Tak zachowuje się typowy program do sondowania sieci.

Żeby sprawdzić, czy masz do czynienia z sondowaniem, musisz jeszcze zajrzeć do odpowiedzi (*opcode 2*), gdyż to one wpisują fałszywy MAC do tablicy ofiary. Spójrz więc na rysunek 4.10, gdzie znajdują się pakiety odfiltrowane filtrem `arp.opcode == 2`.

Rysunek przedstawia masowe *ARP Reply*. Powtarza się w nich, to, że stacja KALI z adresem MAC `00:0c:29:8a:c8:db` rozgłasza, że wiele adresów w podsieci jest przypisanych do adresu MAC stacji KALI. Jest to typowy efekt ataków ARP. Jeśli zastosujesz dodatkowo filtr `arp.src.proto_ipv4 == 192.168.154.131`, pojawią się pakiety, w których pole *Sender Protocol Address* ma wartość `192.168.154.131`. Pamiętaj jednak, że w ataku ARP spoofing nie musi to być rzeczywisty adres IP stacji atakującej, lecz często adres, pod który się ona podszywa, np. adres bramy.

*Ethernet0 2

Plik Edytuj Widok Idź Przechwytyj Analizuj Statystyki Telefonia Bezprzewodowe Narzędzia Pomoc

arp

No.	Time	Source	Destination	Protocol	Length	Info
19534	112.289485	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.123? Tell 192.168.154.131
19535	112.289519	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.191? Tell 192.168.154.131
19536	112.320094	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.124? Tell 192.168.154.131
19537	112.320094	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.193? Tell 192.168.154.131
19538	112.320094	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.194? Tell 192.168.154.131
19539	112.320123	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.122? Tell 192.168.154.131
19540	112.349705	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.195? Tell 192.168.154.131
19541	112.349705	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.196? Tell 192.168.154.131
19542	112.349705	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.125? Tell 192.168.154.131
19545	112.395265	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.128? Tell 192.168.154.131
19546	112.395299	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.126? Tell 192.168.154.131
19547	112.410240	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.129? Tell 192.168.154.131
19548	112.410283	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.198? Tell 192.168.154.131
19549	112.410283	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.130? Tell 192.168.154.131
19550	112.410283	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.197? Tell 192.168.154.131
19551	112.410283	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.127? Tell 192.168.154.131
19552	112.425502	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.200? Tell 192.168.154.131
19553	112.425513	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.133? Tell 192.168.154.131
19554	112.425513	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.199? Tell 192.168.154.131
19555	112.455622	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.202? Tell 192.168.154.131
19556	112.455622	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.203? Tell 192.168.154.131
19557	112.455622	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.135? Tell 192.168.154.131
19558	112.455622	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.201? Tell 192.168.154.131
19559	112.455622	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.134? Tell 192.168.154.131
19560	112.485876	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.205? Tell 192.168.154.131
19561	112.485876	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.137? Tell 192.168.154.131
19562	112.485876	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.136? Tell 192.168.154.131
19563	112.485899	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.204? Tell 192.168.154.131
19564	112.516255	00:0c:29:8a:c8:db	ff:ff:ff:ff:ff:ff	ARP	60	who has 192.168.154.139? Tell 192.168.154.131

> Frame 19534: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NPF_{5D647FA7-2171-49F4-A95E-980DF}

> Ethernet II, Src: 00:0c:29:8a:c8:db, Dst: ff:ff:ff:ff:ff:ff

> Address Resolution Protocol (request)

- Hardware type: Ethernet (1)
- Protocol type: IPv4 (0x0800)
- Hardware size: 6
- Protocol size: 4
- Opcode: request (1)
- Sender MAC address: 00:0c:29:8a:c8:db
- Sender IP address: 192.168.154.131
- Target MAC address: 00:00:00:00:00:00
- Target IP address: 192.168.154.123

RYSUNEK 4.9. Ramki ARP przechwycone w trakcie ataku ARP spoof

Pakiety *ICMP Redirect* nie są typowym ani wymaganym wskaźnikiem ataku ARP spoofing/MiTM. Mechanizm *Redirect* jest związany z decyzjami routingu i jest generowany przez routery w określonych sytuacjach. W analizie ARP MiTM należy koncentrować się przede wszystkim na niespójnych mapowaniach IP/MAC, masowych *ARP Reply*, częstych *gratuitous ARP* oraz na zmianach adresu MAC przypisanego do bramy lub hostów końcowych.

Możesz zastosować filtr `icmp && (ip.src == 192.168.154.132 || ip.dst == 192.168.154.132)`, gdzie podany adres IP to adres stacji atakowanej. Filtr ten może ujawnić dodatkowe komunikaty ICMP związane z zachowaniem routingu w badanym środowisku, ale ich obecność jeszcze nie potwierdza przekazywania ruchu przez host atakujący. Nie jest to jednak ostateczny dowód, bo pakiety *ICMP Redirect* to osobny mechanizm *ICMP (kod 5)* wysyłany przez routery.

*Ethernet0 2

Plik Edytuj Widok Idź Przechwytyj Analizuj Statystyki Telefonia Bezprzewodowe Narzędzia Pomoc

arp.opcode == 2

No.	Time	Source	Destination	Protocol	Length	Info
18903	105.271232	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.227 is at 00:0c:29:8a:c8:db
18904	105.271253	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.228 is at 00:0c:29:8a:c8:db
18905	105.271258	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.229 is at 00:0c:29:8a:c8:db
18906	105.271292	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.230 is at 00:0c:29:8a:c8:db
18907	105.271298	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.231 is at 00:0c:29:8a:c8:db
18908	105.271320	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.232 is at 00:0c:29:8a:c8:db
18909	105.271326	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.233 is at 00:0c:29:8a:c8:db
18910	105.271343	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.234 is at 00:0c:29:8a:c8:db
18911	105.271349	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.235 is at 00:0c:29:8a:c8:db
18912	105.271368	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.236 is at 00:0c:29:8a:c8:db
18913	105.271373	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.237 is at 00:0c:29:8a:c8:db
18914	105.271398	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.238 is at 00:0c:29:8a:c8:db
18915	105.271404	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.239 is at 00:0c:29:8a:c8:db
18916	105.271429	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.240 is at 00:0c:29:8a:c8:db
18917	105.271434	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.241 is at 00:0c:29:8a:c8:db
18918	105.271453	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.242 is at 00:0c:29:8a:c8:db
18919	105.271453	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.243 is at 00:0c:29:8a:c8:db
18920	105.271476	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.244 is at 00:0c:29:8a:c8:db
18921	105.271482	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.245 is at 00:0c:29:8a:c8:db
18922	105.271511	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.246 is at 00:0c:29:8a:c8:db
18923	105.271516	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.247 is at 00:0c:29:8a:c8:db
18924	105.271534	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.248 is at 00:0c:29:8a:c8:db
18925	105.271550	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.249 is at 00:0c:29:8a:c8:db
18926	105.271556	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.250 is at 00:0c:29:8a:c8:db
18927	105.271573	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.251 is at 00:0c:29:8a:c8:db
18928	105.271579	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.252 is at 00:0c:29:8a:c8:db
18929	105.271597	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.253 is at 00:0c:29:8a:c8:db
18930	105.271616	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.254 is at 00:0c:29:8a:c8:db
18931	105.271622	00:0c:29:8a:c8:db	12:34:12:34:12:34	ARP	60	192.168.154.255 is at 00:0c:29:8a:c8:db

> Frame 18903: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NPF_{5D647FA7-2171-49F4-A95E-}

> Ethernet II, Src: 00:0c:29:8a:c8:db, Dst: 12:34:12:34:12:34

▼ Address Resolution Protocol (reply)

Hardware type: Ethernet (1)

Protocol type: IPv4 (0x0800)

Hardware size: 6

Protocol size: 4

Opcode: reply (2)

Sender MAC address: 00:0c:29:8a:c8:db

Sender IP address: 192.168.154.227

Target MAC address: 12:34:12:34:12:34

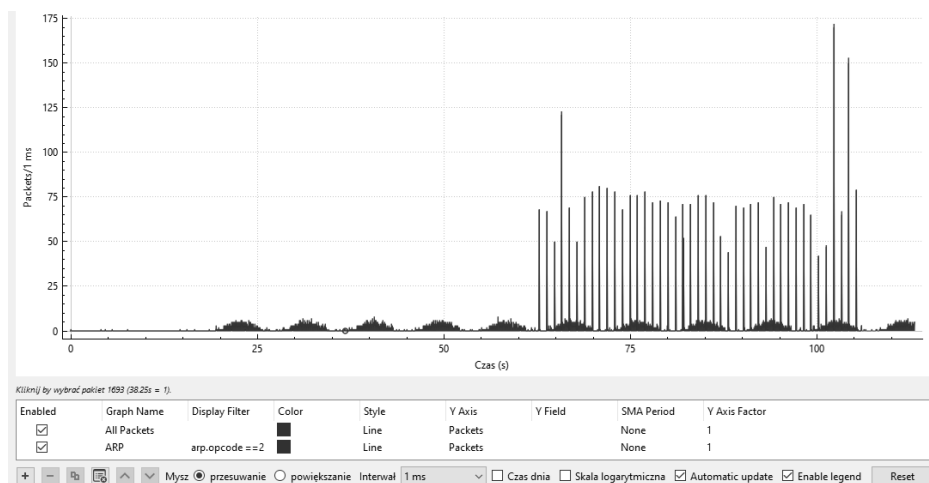
Target IP address: 192.168.154.132

RYSUNEK 4.10. Ramki odpowiedzi ARP

Reasumując: typowy atak ARP najpierw charakteryzuje się nagłym wzrostem liczby *ARP Reply* (*opcode* 2), które są generowane przez jeden adres MAC. Dodatkowo wiele adresów IP zostaje powiązanych z tym samym adresem MAC, co jest jednym z najbardziej charakterystycznych objawów zatrucia tablic ARP.

Jeśli chcesz, możesz od razu zobaczyć to na wykresie. W programie Wireshark wybierz menu *Statystyki*, a potem *I/O graphs*. Następnie ustaw interwał na 1ms, a w filtrze podaj `arp.opcode==2`. Rysunek 4.11 pokazuje gwałtowny skok pakietów ARP. To pokazuje jasno, że coś jest nie tak.

Do wstępnego wychwycenia skanu SYN można użyć filtra `tcp.flags.syn==1 && tcp.flags.ack==0`, a następnie analizować częstotliwość ramek, zakres portów docelowych, liczbę hostów docelowych oraz brak dokończenia połączeń. Jeśli w danym laboratorium występują dodatkowe cechy, takie jak określony rozmiar okna lub układ opcji TCP, można je traktować jako pomocnicze wskaźniki, a nie jako warunek konieczny.



RYSUNEK 4.11. Graf ramek ARP

MAC flooding (przepełnienie tablicy CAM)

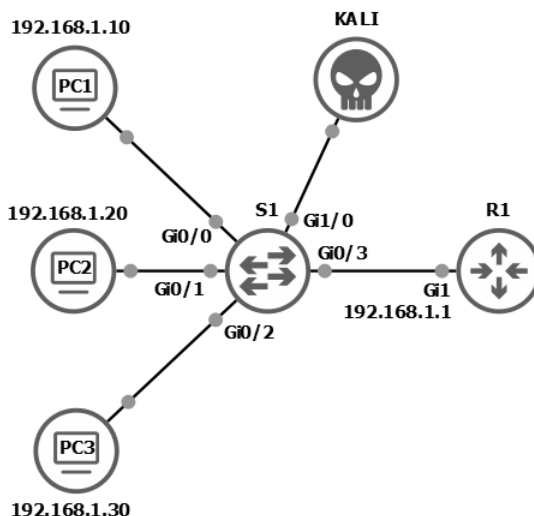
Atak typu *MAC flooding* należy do klasycznych zagrożeń bezpieczeństwa w warstwie drugiej modelu OSI i jest ściśle związany z zasadą działania sieci Ethernet oraz przełączników mostkujących. Szczercie mówiąc, w wielu sieciach, z którymi się spotkałem, ten atak niestety wciąż może się powieść. Ethernet jako technologia warstwy łącza danych jest definiowany przez rodzinę standardów IEEE 802.3, natomiast logika mostkowania, filtrowania i przekazywania ramek między portami przełącznika wynika z mechanizmów opisanych obecnie w rodzinie standardów IEEE 802.1, w praktyce odnoszonych do mostów i sieci mostkowanych. Przełącznik warstwy drugiej podejmuje decyzję o przekazaniu ramki na podstawie adresu MAC odbiorcy oraz własnej tablicy skojarzeń adresów z portami. Jeżeli przełącznik zna port przypisany do danego adresu docelowego, przekazuje ramkę wyłącznie tam, gdzie powinien. Jeżeli jednak adres docelowy nie jest znany, ramka może zostać rozesłana na porty należące do tego samego segmentu logicznego. Właśnie tę cechę próbuje wykorzystać atak *MAC flooding*.

Spójrz na rysunek 4.12. Znajduje się na nim topologia, która posłuży Ci do tego, aby samemu zasymulować tego typu atak, by później przeanalizować przechwycony ruch. W topologii do przełącznika S1 podłączono trzy hosty użytkowników, stację atakującą KALI oraz router R1 pełniący funkcję bramy domyślnej dla sieci 192.168.1.0/24. Hosty PC1, PC2 i PC3 pracują w tej samej domenie rozgłoszeniowej, a ich ruch przechodzi przez centralny przełącznik.

Aby przeprowadzić symulację, a właściwie atak, przejdź do stacji KALI. Do wykonania ataku wykorzystasz narzędzie *macof*, wchodzące w skład pakietu *dsniff*. W oficjalnym opisie dystrybucji Kali Linux narzędzie to zostało scharakteryzowane wprost jako program służący do zalewania sieci lokalnej losowymi adresami MAC. Jego zadaniem nie jest więc prowadzenie zwykłej komunikacji, lecz wygenerowanie bardzo dużej liczby ramek

RYСУNEK 4.12.

Topologia składająca się z trzech stacji roboczych, routera i stacji atakującej KALI



Ethernet z dynamicznie zmieniającymi polami źródłowymi i docelowymi, tak aby przełącznik był zmuszony do ciągłego uczenia się nowych wpisów. Aby rozpocząć symulację, użyj polecenia `sudo macof -i eth0 -n 50000`, które uruchamia generowanie dużej liczby ramek przez interfejs `eth0`. Początek ataku ilustruje rysunek 4.13.

```
(kali@kali)-[~]
└─$ sudo macof -i eth0 -n 50000
[sudo] password for kali:
6c:2c:4e:60:3b:c8 3:ee:c3:66:c6:59 0.0.0.0.58841 > 0.0.0.0.25838: S 227659444:227659444(0) win 512
bf:8f:c6:2:98:5a df:bb:f7:21:ee:85 0.0.0.0.33452 > 0.0.0.0.13056: S 2098391686:2098391686(0) win 512
23:9d:3b:7b:38:cf 46:a2:7e:46:db:ee 0.0.0.0.12839 > 0.0.0.0.29572: S 722459569:722459569(0) win 512
0:7c:44:7a:1d:8 34:ff:99:40:3:6a 0.0.0.0.51746 > 0.0.0.0.18364: S 497931901:497931901(0) win 512
b9:c4:6b:23:3c:66 6f:f3:24:46:99:80 0.0.0.0.35155 > 0.0.0.0.28374: S 1742216045:1742216045(0) win 512
d5:8:a0:75:90:b8 11:43:83:2:1b:ab 0.0.0.0.5271 > 0.0.0.0.55699: S 1582322774:1582322774(0) win 512
8c:10:a0:2e:56:d0 66:f8:e2:5a:c1:f 0.0.0.0.33104 > 0.0.0.0.1148: S 161194502:161194502(0) win 512
53:9d:30:50:bf:66 9:be:ff:18:c3:90 0.0.0.0.31024 > 0.0.0.0.28843: S 1614354698:1614354698(0) win 512
85:e4:a4:7:df:21 2d:4d:54:78:96:27 0.0.0.0.2152 > 0.0.0.0.57746: S 2075053612:2075053612(0) win 512
da:d1:f1:5:b8:de 14:2d:89:70:40:ca 0.0.0.0.9354 > 0.0.0.0.28617: S 1749314539:1749314539(0) win 512
96:c7:f1:6c:3e:9 4f:2d:bb:28:9f:85 0.0.0.0.12243 > 0.0.0.0.4913: S 1503954259:1503954259(0) win 512
```

RYСУNEK 4.13. Atak za pomocą narzędzia `macof` ze stacji KALI

Aby zebrać informacje, jakie są przesyłane podczas ataku, włącz przechwytywanie na łączu między stacją KALI a przełącznikiem S1. Na rysunku 4.14 znajduje się fragment przechwyconego ruchu. Możesz tam zauważyć bardzo dużą liczbę krótkich ramek Ethernet, pojawiających się w niezwykle małych odstępach czasu. Charakterystyczne jest to, że kolejne ramki zawierają stale zmieniające się adresy MAC nadawcy, a często także nietypowe lub losowe adresy IP w polach wyższych warstw. Taki obraz nie odpowiada normalnej komunikacji użytkowej i jeśli coś takiego zobaczysz w rzeczywistej sieci, od razu powinna zapalić Ci się czerwona lampka. W standardowym ruchu stacja robocza lub serwer używa zwykle jednego, stabilnego adresu MAC przypisanego do interfejsu, natomiast na rysunku źródłowe identyfikatory warstwy drugiej zmieniają się lawinowo. Dla analityka jest to podstawowy artefakt wskazujący na próbę przeciążenia procesu uczenia tablicy przełącznika. Istotne znaczenie ma również analiza czasu. Jeżeli w ułamkach sekundy pojawiają się setki kolejnych ramek, oznacza to, że przełącznik otrzymuje ogromną liczbę nowych wartości `eth.src` w bardzo krótkim oknie czasowym, a więc może bardzo szybko utracić zdolność do utrzymywania stabilnych skojarzeń adresów z portami.

No.	Time	DELTA	Source	Destination	Protocol	Length	Info
19	28.091240	2.006817	0c:44:6f:e7:00:04	01:80:c2:00:00:00	STP	60	Conf. Root = 32768/1/0c:44:6f:e7:00:00 Cost = 0 Port = 0x8005
20	29.479968	1.388728	90.114.9.43	31.96.120.35	IPv4	60	
21	29.480052	0.000084	253.244.237.80	186.8.176.6	IPv4	60	
22	29.480076	0.000024	192.157.248.44	123.105.86.104	IPv4	60	
23	29.480099	0.000023	227.97.96.54	143.113.163.107	IPv4	60	
24	29.480125	0.000026	42.236.42.48	109.232.18.23	IPv4	60	
25	29.480153	0.000028	107.167.232.102	209.224.123.47	IPv4	60	
26	29.480207	0.000054	90.58.157.57	73.90.67.17	IPv4	60	
27	29.480243	0.000036	88.231.106.114	51.170.42.73	IPv4	60	
28	29.480265	0.000022	251.204.247.17	153.244.60.95	IPv4	60	
29	29.480284	0.000019	121.238.136.25	110.103.38.42	IPv4	60	
30	29.480325	0.000041	1.154.202.78	168.20.38.27	IPv4	60	
31	29.480343	0.000018	111.145.113.107	183.55.38.23	IPv4	60	
32	29.480358	0.000015	100.181.89.52	243.132.139.40	IPv4	60	
Ethernet II, Src: a2:ba:46:0e:43:55, Dst: a2:d1:f4:20:5f:25							
Destination: a2:d1:f4:20:5f:25							
Source: a2:ba:46:0e:43:55							
Type: IPv4 (0x0800)							
[Stream index: 17]							
Trailer: 5d1633d4317bfce800000000500202008b7c0000000000000000							
Internet Protocol Version 4, Src: 63.74.172.94, Dst: 17.99.102.12							

RYSUNEK 4.14. Ruch przechwycony na łączu KALI-S1

Następny przechwyt, znajdujący się na rysunku 4.15, wykonano na łączu między hostem PC1 a przełącznikiem S1 i właśnie ten punkt obserwacji ma największą wartość poznawczą w przypadku omówienia tego ataku. W normalnie działającej sieci przełączanej host końcowy nie powinien widzieć obcego ruchu unicast, który nie jest do niego adresowany. Tymczasem w przechwyceniu po stronie PC1 pojawiają się ramki IPv4 z adresami źródłowymi i docelowymi niezwiązanymi z komunikacją tej stacji. Z analitycznego punktu widzenia jest to kluczowa przesłanka wskazująca, że przełącznik przestał poprawnie filtrować część ruchu unicast na podstawie znanych wpisów tablicy adresów. Cisco wyjaśnia, że przy nieznanym adresie docelowym ramka jest rozsyłana na wszystkie porty w danym VLAN-ie z wyjątkiem portu wejściowego, a jednym z powodów takiego stanu może być przepełnienie tablicy przekazywania. W praktyce oznacza to, że część ramek zaczyna być traktowana jako unknown unicast i trafia również na porty, które w poprawnie funkcjonującej sieci nie powinny ich otrzymywać. To właśnie tłumaczy obecność obcego ruchu na łączu PC1-S1.

RYSUNEK 4.15.

Ruch przechwycony
na łączu PC1-S1

No.	Time	DELTA	Source	Destination	Protocol	Length	Info
23	25.501009	0.002649	227.97.96.54	143.113.163.107	IPv4	60	
24	25.507191	0.006182	227.97.96.54	143.113.163.107	IPv4	60	
25	25.509673	0.002482	42.236.42.48	109.232.18.23	IPv4	60	
26	25.514368	0.004695	107.167.232.102	209.224.123.47	IPv4	60	
27	25.516426	0.002058	107.167.232.102	209.224.123.47	IPv4	60	
28	25.520197	0.003771	90.58.157.57	73.90.67.17	IPv4	60	
29	25.521710	0.001513	90.58.157.57	73.90.67.17	IPv4	60	
30	25.525199	0.003489	88.231.106.114	51.170.42.73	IPv4	60	
31	25.526833	0.001634	251.204.247.17	153.244.60.95	IPv4	60	
32	25.530790	0.003957	121.238.136.25	110.103.38.42	IPv4	60	
33	25.532806	0.002016	121.238.136.25	110.103.38.42	IPv4	60	
34	25.536395	0.003589	1.154.202.78	168.20.38.27	IPv4	60	
35	25.538427	0.002032	1.154.202.78	168.20.38.27	IPv4	60	
36	25.542459	0.004032	111.145.113.107	183.55.38.23	IPv4	60	
37	25.544414	0.001955	100.181.89.52	243.132.139.40	IPv4	60	
38	25.548748	0.004334	242.65.213.15	67.154.61.71	IPv4	60	
39	25.550917	0.002169	63.74.172.94	17.99.102.12	IPv4	60	
Frame 39: Packet, 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface -, id 0							
Ethernet II, Src: a2:ba:46:0e:43:55, Dst: a2:d1:f4:20:5f:25							
Destination: a2:d1:f4:20:5f:25							
Source: a2:ba:46:0e:43:55							
Type: IPv4 (0x0800)							
[Stream index: 17]							
Trailer: 5d1633d4317bfce800000000500202008b7c0000000000000000							
Internet Protocol Version 4, Src: 63.74.172.94, Dst: 17.99.102.12							

Zapewne zauważysz, że w środowisku laboratoryjnym efekt ataku pojawi się bardzo szybko. Wynika to z samej natury ataku. Przełącznik uczy się adresu źródłowego MAC

z każdej odebranej ramki, dlatego masowe generowanie ramek z losowymi adresami prowadzi do gwałtownego wzrostu liczby potencjalnych wpisów. Jeżeli generator pracuje z dużą intensywnością, to już po krótkim czasie mogą być widoczne objawy utraty selektywnego przełączania. W realnych wdrożeniach wpływ na skalę i tempo zjawiska ma pojemność tablic sprzętowych, architektura przełącznika, mechanizmy ochronne producenta oraz charakterystyka ruchu w danym VLAN-ie, jednak mechanizm analityczny pozostaje ten sam: nagły wzrost liczby źródłowych adresów MAC po stronie atakującego i pojawienie się obcego ruchu unicast po stronie hosta nieuczestniczącego w wymianie danych.

Na koniec chciałbym jeszcze zwrócić uwagę na to, że z perspektywy obronnej najważniejsze jest ograniczenie możliwości niekontrolowanego uczenia się wielu adresów MAC na portach dostępowych oraz utrudnienie rozlewania nieznanego ruchu unicast. W praktyce stosuje się funkcje typu port security, odpowiednią segmentację sieci VLAN, ograniczanie liczby adresów MAC dopuszczanych na porcie, a także mechanizmy wykrywania anomalii w ruchu warstwy drugiej. Istotne znaczenie ma również właściwe projektowanie sieci, ponieważ skutki przepełnienia tablic przełączania są szczególnie niebezpieczne tam, gdzie wiele hostów współdzieli ten sam rozległy segment VLAN-u.

Jeśli podczas analizy ruchu sieciowego w rzeczywistej sieci coś budzi Twoje obawy, wówczas możesz posłużyć się dodatkowymi filtrami, które same w sobie nie wykryją ataku, ale pozwolą Ci upewnić się o jego wystąpieniu.

Najbardziej podstawowy filtr, od którego warto zacząć, to po prostu `eth`. Wprawdzie nie wykrywa ataku, ale porządkuje analizę i pozwala skupić się na warstwie drugiej. W przechwycie z portu atakującego taki filtr szybko ujawnia nienaturalnie dużą liczbę krótkich ramek Ethernet pojawiających się w bardzo małych odstępach czasu.

Jeżeli chcesz pokazać ruch z jednego konkretnego hosta atakującego, bardzo przydatne jest filtrowanie po fizycznym adresie interfejsu tego hosta, o ile w danym scenariuszu chociaż część ramek rzeczywiście wychodzi z jego prawdziwego adresu MAC. W praktyce przy `macof` to nie zawsze będzie najważniejszy trop, bo narzędzie właśnie zmienia pola źródłowe, ale w niektórych przechwyceniach i przy wstępnej orientacji może pomóc filtr `eth.addr == aa:bb:cc:dd:ee:ff` albo precyzyjniej `eth.src == aa:bb:cc:dd:ee:ff`.

Bardzo użyteczny jest filtr odrzucający ruch rozgłoszeniowy i multicastingowy. Dzięki niemu łatwiej zobaczysz nietypowy unicast, który pojawia się na porcie ofiary. Możesz użyć np. `eth.dst != ff:ff:ff:ff:ff:ff`, co pozwala odsiać broadcasty. A jeżeli chcesz jeszcze mocniej zawęzić obraz do ramek unicast, możesz użyć filtru `!(eth.dst[0] & 1)`. Filtr sprawdza najmłodszy bit pierwszego oktetu adresu docelowego MAC. Jeżeli jest wyzerowany, adres jest unicastowy. Taki filtr może być bardzo pomocny, gdyż na porcie pozwala pokazać rzecz najważniejszą: nie chodzi o to, że host widzi broadcast, bo to jest normalne, lecz o to, że zaczyna widzieć cudzy ruch unicast, który normalnie nie powinien do niego trafiać.

Tak więc bardzo dobry filtr do opisu skutków ataku na porcie hosta może wyglądać tak: `!(eth.dst[0] & 1) && !(eth.addr == MAC_PC1)`, gdzie `MAC_PC1` podstawiasz jako rzeczywisty adres MAC komputera PC1. Filtr po prostu pokaże ramki unicast, które nie są ani wysyłane przez stację, ani do niej adresowane. Jeżeli po zastosowaniu takiego filtru nadal widzisz ruch, to dla analityka jest to bardzo mocna przesłanka, że port hosta otrzymuje cudzy unicast, co może wskazywać na zaburzenie pracy przełącznika po ataku *MAC flooding*.

Przydatny może być również filtr na krótkie ramki. Będzie pomocny, bo generatory typu *macof* często produkują masowo niewielkie ramki. Zatem filtr może wyglądać tak: `frame.len == 60` albo szerzej: `frame.len <= 64`. Pamiętaj jednak, że krótkie ramki same w sobie nie są dowodem ataku, ale w połączeniu z ogromną liczbą zmieniających się adresów źródłowych MAC oraz bardzo małymi odstępami czasu mogą być użytecznym wskaźnikiem. Nie zapomnij o obserwacji kolumny *DELTA* (albo wykresów I/O). Po zastosowaniu filtrów takich jak `eth` albo `eth.type == 0x0800` powinieneś zwracać uwagę na gwałtowne serie ramek pojawiających się w odstępach liczonych w tysięcznych częściach sekundy. To jest w praktyce ważniejsze niż pojedynczy pakiet.

Chciałbym zwrócić uwagę jeszcze na to, że *MAC flooding* nie ma jednego uniwersalnego filtru wykrywającego atak w sposób bezbłędny. Wykrycie opiera się na korelacji kilku symptomów: bardzo dużej liczby ramek w krótkim czasie, szybkich zmian adresów *eth.src*, nietypowej liczby krótkich ramek oraz pojawienia się obcego ruchu unicast na portach, które normalnie nie powinny go odbierać. Zresztą, jak zapewne już zauważyłeś, powtarzam to przy okazji większości analiz.

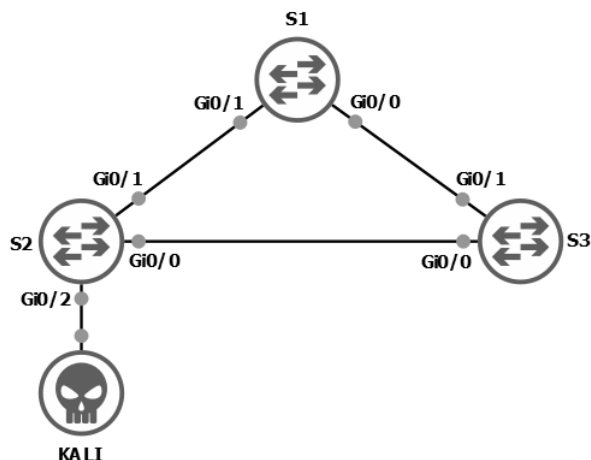
Wykrywanie STP manipulation

Już wiesz, że *Spanning Tree Protocol* należy do podstawowych mechanizmów ochrony topologii przełączanej przed pętlami w warstwie drugiej. Jego zadaniem jest wybór jednego przełącznika pełniącego funkcję root bridge, a następnie wyznaczenie takiej struktury aktywnych połączeń, aby przełączana domena pozostała spójna i jednocześnie wolna od pętli.

W praktyce analitycznej oznacza to, że każda nieautoryzowana zmiana roota lub nagła zmiana parametrów BPDU powinna być traktowana jako zdarzenie potencjalnie niebezpieczne. Mechanizm działania STP i semantyka BPDU wynikają ze standardów IEEE dla mostkowania i drzewa rozpinającego, przede wszystkim z rodziny IEEE 802.1, historycznie kojarzonej z IEEE 802.1D. W odróżnieniu od wielu protokołów warstwy trzeciej i wyższych, STP nie jest klasycznie opisany przez RFC IETF, lecz przez standardy IEEE. Dokumenty IEEE wskazują, że mosty wymieniają BPDU zawierające informacje identyfikujące root bridge, koszt ścieżki do roota oraz identyfikator mostu nadającego wiadomość, a właśnie te pola są dla analityka najważniejsze.

Zacznijmy więc od utworzenia topologii przedstawionej na rysunku 4.16, abyś mógł najpierw przeprowadzić atak, a potem przechwycić ruch i nauczyć się rozpoznawać jego oznaki. W badanym środowisku trzy przełączniki, oznaczone jako S1, S2 i S3, tworzą trójkąt połączeń warstwy drugiej, natomiast do przełącznika S2 został dołączony host KALI. Taki układ jest bardzo dobry do analizy bezpieczeństwa STP, bo pozwala nie tylko wywołać zmianę w logice drzewa rozpinającego, ale również zaobserwować jej skutki na innym łączy niż to, którym host atakujący jest bezpośrednio wpięty do sieci. Z punktu widzenia analityka ma to duże znaczenie, ponieważ w rzeczywistym środowisku przechwyt bardzo często wykonuje się właśnie z miejsca pośredniego, a nie bezpośrednio na porcie napastnika. Przełączniki nie wymagają konfiguracji. Wystarczy, że podłączysz stację KALI i rozpoczniesz atak.

RYSUNEK 4.16.
Topologia z trzema
przełącznikami



W pierwszej kolejności sprawdzimy, który przełącznik pełni funkcję root bridge przed rozpoczęciem ataku. W tym celu na przełączniku S2 wykonaj polecenie `show spanning-tree`. W przedstawionym przykładzie wynik wskazuje jednoznacznie, że to właśnie S2 jest rootem dla VLAN 1. Widać to po zgodności pola `Root ID` z własnym `Bridge ID` przełącznika oraz po komunikacie, że bieżący most jest rootem. Zauważ również, że interfejs `Gi0/2` prowadzący do stacji KALI jest w trybie `Desg`.

W stabilnej domenie STP wszystkie przełączniki powinny propagować w BPDU informację o tym samym root bridge. Jeżeli więc root zostaje nagle zmieniony bez planowanej rekonfiguracji, analityk powinien potraktować to jako zdarzenie podejrzane. Mechanizm działania STP oraz zawartość BPDU są definiowane przez standardy IEEE z rodziny 802.1, historycznie kojarzone z IEEE 802.1D. To właśnie te standardy opisują wybór root bridge, obliczanie kosztów ścieżek i role portów w drzewie rozpinającym.

```

S2#show spann
VLAN0001
Spanning tree enabled protocol ieee
Root ID    Priority    32769
           Address    0c7b.930c.0000
           This bridge is the root
           Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec
Bridge ID  Priority    32769 (priority 32768 sys-id-ext 1)
           Address    0c7b.930c.0000
           Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec
           Aging Time  15 sec

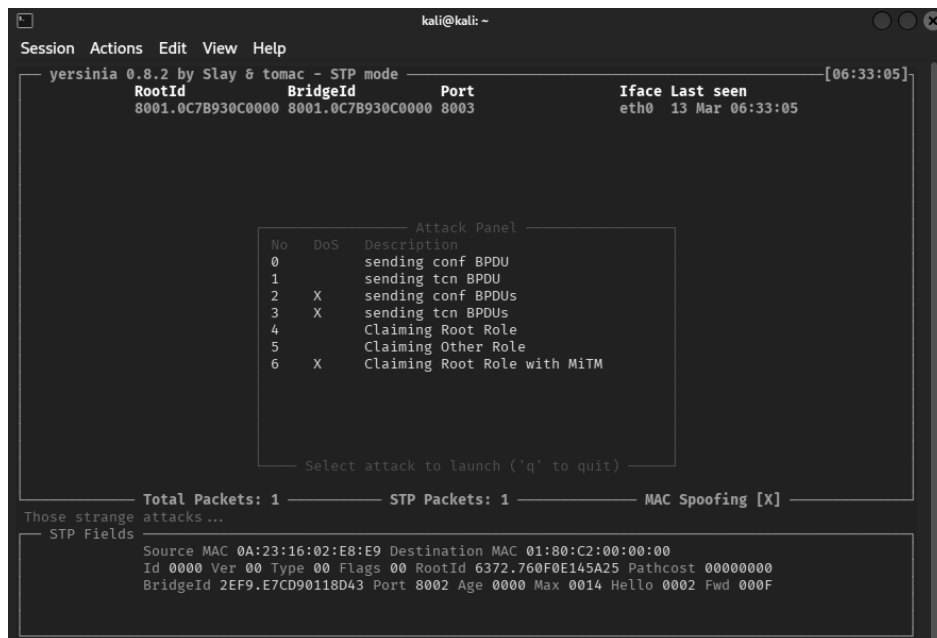
```

Interface	Role	Sts	Cost	Prio.	Nbr	Type
Gi0/0	Desg	FWD	4	128.1		P2p
Gi0/1	Desg	FWD	4	128.2		P2p
Gi0/2	Desg	FWD	4	128.3		P2p

Gdy stan początkowy został już potwierdzony, można przejść do symulacji ataku. Celem włamywacza w takim przypadku jest zrobić wszystko, aby jego stacja została uznana za Root Bridge.

W laboratorium wykorzystasz narzędzie `yersinia`, uruchomione poleceniem `sudo yersinia -I` (rysunek 4.17). W trybie interaktywnym naciśnij `g`, aby wybrać protokół (wybierz protokół STP). Jeśli chcesz, możesz zmienić parametry ramek wysyłanych podczas ataku.

Opcje te zmienisz po naciśnięciu litery *e* (*edit*). Następnie na dole zmień to, co chcesz zmienić. Teraz kliknij literę *x*, aby otworzyć menu ataków (*Execute attack*), i w menu wybierz opcję 4 (*Claiming Root Role*). Sens tego działania polega na wysyłaniu spreparowanych ramek BPDU, które mają sprawić, że legalny przełącznik uzna host atakujący za bardziej korzystnego kandydata na root bridge. Z perspektywy administracyjnej oznacza to próbę przejęcia kontroli nad logiką drzewa rozpinającego, natomiast z perspektywy analityka najważniejsze jest to, że taka operacja powinna pozostawić ślady zarówno w stanie przełączników, jak i w przechwyconym ruchu STP. Zanim rozpoczniiesz atak, włącz przechwytywanie ruchu na łączu S1-S3.



RYSUNEK 4.17. Panel służący do wyboru ataku

Po uruchomieniu ataku sprawdzimy najpierw ponownie stan przełącznika S2 za pomocą tego samego polecenia `show spanning-tree`. Zauważ, że poniższy listing przedstawia teraz inny wynik. S2 nie jest już rootem. Pole `Root ID` wskazuje nowy identyfikator mostu, a port prowadzący do hosta KALI staje się root portem. Oznacza to, że przełącznik S2 zaakceptował informację o nowym root bridge i zaczął traktować ścieżkę prowadzącą przez host końcowy jako najlepszą drogę do roota. W normalnej sytuacji host użytkownika nie powinien mieć możliwości przejęcia roli root bridge w domenie przełączanej. Jeżeli więc port, do którego podłączony jest zwykły host, nagle staje się root portem przełącznika infrastrukturalnego, to mamy do czynienia z objawem klasycznej manipulacji STP.

```
S2#sh spann
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID    Priority    32769
            Address     0c7b.930b.0000
            Cost        4
            Port        3 (GigabitEthernet0/2)
```

Bridge ID	Hello Time	2 sec	Max Age	20 sec	Forward Delay	15 sec
	Priority	32769 (priority	32768	sys-id-ext	1)	
	Address	0c7b.930c.0000				
	Hello Time	2 sec	Max Age	20 sec	Forward Delay	15 sec
	Aging Time	15 sec				
Interface	Role	Sts	Cost	Prio.	Nbr	Type
Gi0/0	Desg	FWD	4	128.1		P2p
Gi0/1	Desg	FWD	4	128.2		P2p
Gi0/2	Root	FWD	4	128.3		P2p
Gi0/3	Desg	FWD	4	128.4		P2p

Na tym etapie można by uznać, że atak został potwierdzony, ale z punktu widzenia analityki ruchu sieciowego najciekawsze dopiero przed nami. Sprawdźmy teraz przechwyt wykonany nie na porcie KALI ani nawet nie na łączu S2–KALI, lecz na łączu pomiędzy S1 i S3. Taki wybór miejsca obserwacji jest celowy, ponieważ pozwala odpowiedzieć na ważne pytanie: czy analityk, który nie widzi bezpośrednio ruchu napastnika, nadal jest w stanie wykryć, że w domenie STP dzieje się coś nieprawidłowego? Odpowiedź brzmi: tak, ale wykrywa on przede wszystkim skutek ataku, a nie sam moment jego wstrzyknięcia.

Rysunek 4.18 przedstawia przechwycone dane. W pierwszej kolejności warto zawęzić analizę do ramek STP, np. filtrem `stp`. Warto też wyświetlić w osobnych kolumnach pola związane z BPDU, przede wszystkim `stp.root.hw`, `stp.bridge.hw`, `stp.root.path_cost` oraz `stp.flags.tc`. Dzięki temu jako analityk nie będziesz musiał za każdym razem rozwijać całego drzewa dekodowania pakietu. Od razu zobaczysz najważniejsze wartości na liście ramek. W przechwycie z łącza S1–S3 widać, że ramki STP są nadal wysyłane na właściwy adres grupowy 01:80:c2:00:00:00, a więc z formalnego punktu widzenia format komunikacji kontrolnej jest poprawny. Problem ujawnia się dopiero po analizie zawartości BPDU.

The screenshot shows a Wireshark capture of STP frames. The packet list on the left shows several frames of type 0 (BPDUs). The details pane on the right is expanded for one of these frames, showing the following information:

- Frame 25: Packet, 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface -, id 0
- IEEE 802.3 Ethernet
- Logical-Link Control
- Spanning Tree Protocol
 - Protocol Identifier: Spanning Tree Protocol (0x0000)
 - Protocol Version Identifier: Spanning Tree (0)
 - BPDU Type: Configuration (0x00)
 - BPDU flags: 0x01, Topology Change
 - 0... .. = Topology Change Acknowledgment: No
 - 0... .. = Topology Change: Yes
 - Root Identifier: 32768 / 1 / 0c:7b:93:0c:00:00
 - Root Path Cost: 8
 - Bridge Identifier: 32768 / 1 / 0c:85:d9:f9:00:00
 - Port Identifier: 0x8002
 - Message Age: 2
 - Max Age: 20
 - Hello Time: 2
 - Forward Delay: 15
 - Originating VLAN (PVST): 1

RYSUNEK 4.18. Ruch STP przechwycony na łączu S1–S3

Zwróć uwagę na pole *Root Identifier*. To właśnie ono ma tutaj znaczenie kluczowe. Jeżeli przed atakiem domena STP wskazywała jednego roota, a po ataku w przechwycie na innym łączu zaczyna pojawiać się inny *Root Identifier*, to znaczy, że zmiana została już przyjęta i jest dalej propagowana przez legalne przełączniki. W omawianym laboratorium jest to dokładnie to, co obserwujemy. Przełączniki nie ogłaszają już poprzedniego, legalnego roota,

lecz nowy identyfikator. Zauważ, że nie widać tu jeszcze yersinii ani bezpośrednio hosta KALI wysyłającego sfałszowane BPDU, ale widać coś równie ważnego, czyli skuteczną zmianę stanu drzewa rozpinającego.

Dalej zwróć uwagę na pole *Bridge Identifier*. Mówi ono, który przełącznik faktycznie wysłał daną BPDU. Jeżeli więc w przechwycie pojawia się sytuacja, w której *Bridge Identifier* należy do legalnego przełącznika, ale *Root Identifier* wskazuje już inny, nowy most, oznacza to, że przełącznik infrastrukturalny sam rozsyła dalej informację o nowym root bridge. To właśnie jest najważniejszy objaw pośredni. Innymi słowy: legalna infrastruktura została zmuszona do uznania nowego roota i zaczęła sama propagować tę zmianę. Z punktu widzenia wykrywania włamań jest to bardzo cenna obserwacja, bo pozwala wykazać incydent nawet wtedy, gdy nie mamy przechwyty z punktu wejścia ataku. Warto również przeanalizować pole *Root Path Cost*. Jeżeli po zmianie roota kolejne przełączniki zaczynają reklamować nowy koszt dojścia do roota, a wartość ta jest logicznie zgodna z topologią, oznacza to, że zaszła rzeczywista rekonwergencja STP. W naszym przypadku widać właśnie taki efekt. Przełącznik znajdujący się bliżej nowego roota raportuje niższy koszt, natomiast urządzenia położone dalej propagują już odpowiednio większą wartość. Dla analityka jest to potwierdzenie, że nie ma do czynienia z pojedynczą anomalią jednej ramki, lecz z realną zmianą w drzewie rozpinającym.

Dodatkową wskazówką może być flaga *Topology Change*. Sama jej obecność nie oznacza jeszcze włamania, ponieważ może pojawiać się również podczas legalnych zmian w topologii. Jeżeli jednak pojawia się równocześnie ze zmianą *Root Identifier*, zmianą ról portów oraz utratą pozycji root bridge przez dotychczasowy przełącznik nadrzędny, to staje się bardzo istotnym elementem korelacji. Analityk nie powinien więc traktować jednej wartości w izolacji, lecz zestawiać razem kilka obserwacji, czyli zmianę roota, zmianę kosztów, zmianę ról portów oraz obecność komunikatów o zmianie topologii.

W tym miejscu chciałbym bardzo wyraźnie podkreślić jedną rzecz. Z przechwyty wykonanego na łączu S1–S3 nie da się zwykle bezpośrednio udowodnić, że to konkretnie host KALI był źródłem pierwotnych sfałszowanych BPDU. Do tego potrzebny byłby przechwyt bliżej portu S2–KALI albo zapis z mechanizmów ochronnych samego przełącznika. Z tego miejsca obserwacji można jednak z dużą pewnością wykazać, że domena STP zmieniła roota w sposób niezgodny z wcześniejszym stanem sieci i że zmiana ta została rozpropagowana przez legalną infrastrukturę. To rozróżnienie jest bardzo ważne. Często w praktyce nie zawsze będziesz identyfikować od razu źródło ataku, ale możesz bardzo skutecznie wykryć jego skutek.

Jeszcze słowo na temat bezpieczeństwa. Aby ograniczyć ryzyko takich incydentów, stosuje się mechanizmy ochronne zapobiegające akceptowaniu BPDU z portów dostępowych oraz wymuszające przewidywalną pozycję legalnych przełączników w drzewie STP. Z punktu widzenia bezpieczeństwa nie chodzi tu tylko o stabilność sieci, ale również o ochronę przed przejęciem wpływu na ścieżki ruchu w warstwie drugiej. Nawet bez wchodzenia w szczegóły konfiguracji warto pamiętać, że dobrze zaprojektowana infrastruktura nie powinna pozwalać zwykłemu hostowi na zmianę roota całej domeny przełączanej.

Podsumowując: w wykrywaniu manipulacji STP kluczowe jest porównanie stanu przed zdarzeniem i po zdarzeniu. Najpierw należy ustalić legalnego roota, następnie sprawdzić, czy po ataku jego rola została przejęta przez inne urządzenie, a na końcu potwierdzić w ewentualnym przechwycie, że nowy *Root Identifier* jest propagowany przez legalne przełączniki.

Użyteczne filtry

Na zakończenie warto poznać jeszcze kilka dodatkowych filtrów, które mogą istotnie ułatwić pracę analityka. Nie są one samodzielnym dowodem ataku, ale w praktyce często pomagają szybciej zawęzić obszar analizy i potwierdzić podejrzenia wynikające z obserwacji ruchu.

Filtr `arp.isgratuitous == 1` pozwala odszukać pakiety *gratuitous ARP*, czyli takie, w których host sam ogłasza mapowanie własnego adresu IP na adres MAC. W normalnej sieci takie ramki mogą pojawiać się legalnie, np. podczas przełączeń awaryjnych lub odświeżania informacji sąsiadów, jednak ich nagłe i częste występowanie bywa bardzo cenną wskazówką przy analizie ARP spoofing. Wireshark udostępnia dla tego zjawiska osobne pole filtru.

Filtr `arp.isprobe == 1` jest przydatny wtedy, gdy chcesz oddzielić zwykłe zapytania ARP od ramek związanych z wykrywaniem konfliktu adresów IPv4. Taki filtr może pomóc odróżnić normalne zachowanie hosta uruchamiającego mechanizm sprawdzania adresu od właściwego etapu zatruwania tablic ARP. Jest to szczególnie użyteczne w środowiskach laboratoryjnych, w których wiele maszyn jest często uruchamianych i wyłączanych. Pole to znajduje się w referencji filtrów Wiresharka dla ARP.

Filtr `arp.duplicate-address-detected` może być bardzo pomocny podczas szukania niespójności w mapowaniach IPv4/MAC. Wireshark potrafi oznaczyć sytuacje sugerujące wcześniejsze użycie tego samego adresu IP i właśnie ten filtr pozwala takie pakiety szybko wyłapać. Nie zastępuje on pełnej analizy incydentu, ale dobrze nadaje się do wstępnego wychwycenia konfliktów, które mogą mieć związek z błędną konfiguracją albo aktywnością atakującego. To pole jest opisane w referencji filtrów ARP Wiresharka.

Filtr `arp.dst.proto_ipv4 == 192.168.1.1` pozwala zawęzić widok do ramek ARP, których celem jest konkretny adres IPv4. W praktyce jest to bardzo wygodne, gdy chcesz sprawdzić, czy wiele stacji nagle zaczęło pytać o adres bramy domyślnej albo wybranego hosta. Taki filtr dobrze uzupełnia analizę ARP, bo zamiast patrzeć wyłącznie na nadawcę, możesz od razu skupić się na ofierze lub na adresie, pod który ktoś się podszywa. Pole `arp.dst.proto_ipv4` jest dostępne w oficjalnej referencji Wiresharka.

Filtr `tcp.flags.reset == 1` może być bardzo pomocny przy analizie skanowania TCP. W rozdziale omówiłem już filtrowanie pakietów SYN, jednak równie wartościowe bywa spojrzenie od drugiej strony, czyli na odpowiedzi resetujące. W przypadku intensywnego skanowania portów zamkniętych analityk może zobaczyć wyraźne serie ramek RST lub RST/ACK, które dobrze pokazują, jak wiele prób dotarło do nieaktywnych usług. Pole `tcp.flags.reset` znajduje się w oficjalnej referencji filtrów TCP Wiresharka.

Filtr `icmp.type == 3 && icmp.code == 3` pozwala od razu wyświetlić komunikaty *ICMP Destination unreachable/Port unreachable*. Jest to bardzo użyteczne przy analizie skanowania UDP, ponieważ takie odpowiedzi często stanowią podstawę do wnioskowania, że port UDP jest zamknięty. Zamiast przeglądać cały ruch ICMP, możesz od razu ograniczyć widok do najważniejszego kodu diagnostycznego. Ten filtr dobrze porządkuje analizę wtedy, gdy przechwyt zawiera wiele różnych komunikatów ICMP. Opis składni filtrów znajduje się w dokumentacji Wiresharka.

Filtr `udp.dstport == 53` (albo analogicznie inny wybrany port UDP) jest przydatny wtedy, gdy chcesz sprawdzić, czy odpytywanie nie koncentrowało się na konkretnych usługach. W praktyce analitycznej pozwala to szybko odróżnić szeroki skan całego zakresu

portów od bardziej ukierunkowanego działania, np. tylko wobec DNS, SNMP czy NTP. Taki filtr nie wykrywa skanu, ale świetnie wspiera analizę intencji intruza i zakresu jego rozpoznania. Referencja pól filtrów TCP i UDP jest dostępna w dokumentacji Wiresharka.

Filtr `stp.flags.tc == 1` pozwala wyłapać ramki BPDU, w których ustawiono znacznik *Topology Change*. W normalnej pracy sieci jego obecność nie musi oznaczać ataku, ale jeśli pojawia się nagle i jednocześnie z innymi oznakami rekonwergencji, może być bardzo ważnym elementem korelacji.

Filtr `stp.root.hw == xx:xx:xx:xx:xx:xx` jest bardzo praktyczny wtedy, gdy chcesz sprawdzić, które ramki reklamują konkretnego roota. Dzięki temu można szybko zobaczyć moment, w którym legalna infrastruktura zaczyna propagować już nowy identyfikator root bridge.

Skorowidz

A

- ACK, ACKnowledgment, 40, 481
- adres
 - IPv4
 - wykrywanie duplikatów, 186
 - IPv6
 - wykrywanie duplikatów, 189
 - MAC, 483
 - czas życia wpisu, 143
 - rozwiązywanie nazw, 292
 - rozgłoszeniowy, broadcast, 145, 160
- adresowanie MAC, 142
- agregacja łączy
 - EtherChannel, 148
 - LACP, 148
- algorytmy podpisów, 100
- ALPN, Application-Layer Protocol Negotiation, 95, 100, 103, 481
- analiza
 - anomalii DNS, 311
 - czasowa i wolumenowa, 422
 - czasu odpowiedzi, 452
 - dostarczania pakietów
 - out-of-order, 86
 - path asymmetry, 84
 - duplikatów ACK, 83
 - Hop Limit, 182
 - HTTP/3, 364
 - incydentów w OSPF, 278
 - jakości rozmowy, 401
 - komunikacji
 - FTP, 328
 - TFTP, 331
 - komunikatów
 - Client Hello, 99, 100
 - Server Hello, 101
 - odpowiedzi DNS, 35
 - ograniczenia przepustowości, 86
 - pakietów IP, 181
 - polecenia traceroute, 184
 - problemów routingu, 227
 - procesu DNS, 32
 - protokołu
 - CDP, 137
 - LLDP, 140
 - QUIC, 358, 364
 - SIP, 382
 - STP, 130
 - routingu, 222
 - rozszyfrowanego ruchu, 107
 - RTT, 58
 - ruchu
 - DNS, 299, 310
 - HTTP, 418
 - sieci VLAN, 133
 - VoIP, 382–385
 - rytmu sieci, 243
 - sesji
 - pocztowych, 345
 - TLS, 93, 103
 - zdalnych, 89
 - statystyk, 54
 - strat i fragmentacji, 465
 - strumienia RTP, 117, 406, 409
 - tablic MAC, 142
 - TCP na podstawie wykresów, 436
 - three-way handshake, 42
 - TTL, 182
 - usługi DNS, 294–345
 - włamań
 - w warstwie drugiej OSI, 158
 - w warstwie trzeciej OSI, 235
 - wydajności, 450
 - czas odpowiedzi, 452
 - narzut nagłówków HTTP, 460, 465
 - objętość odpowiedzi, 462, 465
 - rozbudowane ciasteczka, 463
 - statystyka żądań HTTP, 459
 - zjawisko gadatliwej aplikacji, 454
- anomalie, 436
 - błędy uwierzytelnienia, 290
 - DNS, 311
 - interpretacja, 259
 - przechwytywanie hasła OSPF, 289

anomalie

TTL, 259

w zestawianiu sąsiedztwa OSPF, 279

wykrywanie, 82, 274

ARP, 481

Announcement, 165

poisoning, 165

Probe, 165

Reply, 167

Request, 167

Asterisk

instalacja, 388

konfiguracja, 389

modyfikowanie konfiguracji, 393

atak

ARP spoofing, 165, 167, 324

ARP spoofing/MiTM, 168

brute force, 114, 356

brute force na OSPF, 291

cache poisoning, 304, 481

DHCP Starvation, 315

ICMP error abuse, 246

ICMP reconnaissance, 241

IP spoofing, 238

MAC flooding, 170

MiTM, man-in-the-middle, 117, 165, 324, 325

na pocztę elektroniczną, 350

na IMAPS, 353

na SMTP, 356

phishing, 351

na routing OSPF, 278

Overlapping Fragments, 221

ping flood, 58

połączeniowy, 274

Teardrop, 221

typu flood, 416

typu reconnaissance, 256

typu rogue DHCP, 311

typu TTL manipulation, 259

związany z tunelowaniem, 274

awarie, 116

B

biblioteka

npcap, 11

Scapy, 254

big-endian, 69

bilet, Session Ticket, 98

Bloki adresów MAC, 155, 156

błąd bad record MAC, 110

błędy TCP, 58

BPDU, 481

burza rozgłoszeniowa, broadcast storm, 126 481

C

cache poisoning, 304, 481

capture filter, 481

CDP, Cisco Discovery Protocol, 481

certyfikat samopodpisany, self-signed, 102, 339

ciasteczka, 463

Cisco Discovery Protocol

Addresses, 139

Device ID, 138

Platform, 139

Software Version, 139

Client Hello, 96-100, 366, 481

connection pooling, 43

Conversations, 482

Ethernet, 128

Identyfikator strumienia, 425, 428

IPv4, 128

komunikacja do adresu grupowego, 292

Podążaj za strumieniem..., 425

statystyki komunikacji tunelowej, 276

TCP, 425

D

dane geolokalizacyjne GeoLite2, 209

datagram UDP, 254

DBD, Database Description, 282

DHCP, 231

Discover, 233

fałszywy serwer, 311

mechanizm DHCP Snooping, 311

mechanizm port security, 311

Offer, 232

Request, 232

display filter, 482

Długości pakietu, Packet Lengths, 57

DNS, Domain Name System, 30, 294-345

błędne odpowiedzi, 301

błędy przetwarzania, 300

jako źródło danych, 307

kod odpowiedzi

NODATA, 302, 306

NOERROR, 310

NXDOMAIN, 301, 306, 310, 484

REFUSED, 300, 310, 484

SERVFAIL, 300, 310, 485

limit timeout serwera, 299

lista zapytań, 296

nietypowe typy operacji, 304

równoległe rozwiązywanie nazw, 305

standardowa komunikacja, 295

symulacja opóźnień, 297

zatrucie pamięci podręcznej, 304

DNSSEC, 482

dokumenty RFC, 9

Domain Name System (query)

flaga Non-authenticated data, 34

flaga Opcode, 33

flaga Recursion, 34

flaga Truncated, 34

Flags, 34

Label count, 34

Name Length, 34

Queries, 34

Question, 34

RCODE, 33

Transaction ID, 33

Domain Name System (response)

Answers, 35

Authority, 301

DNS ID, 296, 298, 303

DNS-Time, 298, 299, 304

Opcode, 301

Recursion available, 301

Reply code, 301

Response, 301

Dovecot, 347

DSAP, Destination Service Access Point, 147, 482

DTP, Dynamic Trunking Protocol, 147, 482

Duplicate ACK, 79, 200, 201, 204, 482

dyscyplina kolejkowania, queueing discipline, 197
działanie

komunikacji w DNS, 30

mechanizmu three-way handshake, 40

protokołu

ARP, 51

DHCP, 231

STP, 124

przełącznika, 143

E

ECDHE, 96, 97, 100–102, 109

ECH, Encrypted Client Hello, 94, 367, 482

Edycja

Preferencje/Kolumny, 47

eksport

kluczy sesji TLS, 28

pakietów, 24–27

Eksportuj wybrane pakiety

Zachowaj, 25

Zakres pakietów, 25

Endpoints

analizowanie ruchu ICMP, 245

Kopiuj, 210

Map, 210

EtherChannel, 148, 151, 152, 482

Ethernet II, 482

Expert Information, 82, 444, 482

F

fast retransmission, 482

FDB, Filtering Database, 143

FDB, Forwarding Database, 482

Filtr konwersacji, 53

Ethernet, 53

IPv6, 53

Filtr przechwytywania, capture filter, 19, 113, 481

Filtr przechwytywania dla wybranych
interfejsów, 19

Filtr wyświetlania, display filter, 16, 37, 113, 482
tworzenie, 39

tworzenie przycisku, 40

filtr

!(eth.dst[0] & 1) && !(eth.addr == []), 173

!(stp or lldp or cdp or lacp or arp), 157

!(tcp.port == 443), 115

!(tcp.stream == 215), 427

.analysis.window_update, 432

arp, 51, 144

arp.dst.proto_ipv4 == [], 179

arp.duplicate-address-detected, 179

arp.isgratuitous == 1, 179

arp.isprobe == 1, 179

arp.opcode == 2, 73, 154

cdp, 138, 139, 154

dhcp, 127, 128, 232, 314

dns, 297, 307

dns && dns.qry.name == "wikipedia.org",
296, 297, 303

dns && dns.qry.name == [], 299

dns && dns.qry.type == 16, 303

dns or icmp, 36

dns.a, 121

dns.flags.opcode != 0, 311

dns.flags.rcode == 5, 311

dns.flags.response == 1, 120, 311

dns.flags.response == 1 && dns.flags.rcode == 0
&& dns.count.answers == 0, 311

dns.qry.name == "helion.pl", 32

eth, 153, 173

eth.addr == aa:bb:cc:dd:ee:ff, 157, 173

eth.addr[0] & 1, 153

eth.dst != ff:ff:ff:ff:ff:ff and not eth.dst[0] & 1, 144

eth.dst == aa:bb:cc:dd:ee:ff, 153, 157

eth.dst[0] & 1 == 0, 144

eth.dst[0] & 1 == 0 and not eth.dst == [], 144

eth.dst == ff:ff:ff:ff:ff:ff, 128, 153

eth.src == aa:bb:cc:dd:ee:ff, 153, 156, 173

eth.type == 0x0800, 157

frame contains "...", 74

frame.len < 64, 157

frame.len > 1500, 121

frame.number == 122 || frame.number == 127,
81

filtr

```

frame.protocols contains "vlan,vlan", 92
ftp, 330
gre.proto == 0x0800, 293
http, 48, 418, 456, 458
http.content_length > 1000000, 480
http.request, 74
http.request && !http.host contains "...", 480
http.request.method == "POST" &&
    http.request.uri contains "login", 115
http.response, 74
http.response.code == 304, 480
http.response.code == 401, 114
http2, 105
http2.header.value contains
    "youtubei/v1/player", 107
http2.streamid == 61, 108
http3, 361
http3.frame_type == 0x0c, 365
icmp && ip.addr == [] && ip.addr == [], 218
icmp, 19, 237, 245
icmp.type == 11, 120
icmp.type == 3, 120, 233, 250
icmp.type == 3 && icmp.code == 3, 179, 248
icmp.type == 3 && icmp.code == 4, 219, 480
icmpv6.type == 1, 233
icmpv6.type == 2, 220
ip or ipv6 and frame.comment contains
    "Fragmented", 220
ip.addr == [], 37, 425
ip.addr == [] && !(tcp.stream == N), 426
ip.addr == [] && icmp, 293
ip.addr == [] && tcp.analysis.retransmission,
    114
ip.addr == [] && tcp.port == 443, 116
ip.dst ==, 423
ip.flags.df == 1 && ip.len > 1400, 480
ip.flags.mf == 1 && ip.flags.df == 1, 293
ip.flags.mf == 1 || ip.frag_offset > 0, 220, 233
ip.frag_offset > 0, 293
ip.id == 12345, 255
ip.id == 0 && ip.flags.df == 0, 293
ip.id == 0x0000, 225
ip.proto == 4, 268
ip.src == ip.dst, 293, 422
ip.ttl < 5, 233, 293
ipv6.hlim < 5, 233
ipv6.nxt == 44, 221
lACP, 151, 154
lldp, 140, 154
not vlan, 92, 157
ospf, 228, 257, 279, 290
ospf.db.interface_mtu > 1500, 284, 293
ospf.hello.hello_interval < 10, 229
ospf.hello.router_dead_interval > 40, 229
ospf.msg == [], 234, 284

```

```

ospf.msg == 4 or ospf.msg == 5, 230
quic, 361
rtcp, 414
rtcp.ssrc.fraction > 0, 414
rtcp.ssrc.jitter > 0, 414
rtp.ssrc == 0x..., 413
sdp.media.port, 413
sdp.media.proto, 413
sip && sdp, 413
sip.CSeq.method == "BYE", 413
srtp, 414
ssh, 319
stp, 177
stp.flags.tc == 1, 180
stp.root.hw == xx:xx:xx:xx:xx:xx, 180
stp.version == [], 154
tcp port 443, 113
tcp.analysis.ack_rtt > 0.1, 432, 435
tcp.analysis.duplicate_ack, 73, 86, 120, 440
tcp.analysis.duplicate_ack && tcp.stream == [],
    432, 434
tcp.analysis.duplicate_ack ||
    tcp.analysis.retransmission, 81
tcp.analysis.fast_retransmission, 81, 86, 234
tcp.analysis.flags, 86
tcp.analysis.flags && !tcp.analysis.win-
    dow_update, 480
tcp.analysis.keep_alive, 480
tcp.analysis.out_of_order, 73, 81, 86, 113, 234
tcp.analysis.retransmission, 73, 86, 98, 113,
    234, 379, 440
tcp.analysis.retransmission && tcp.stream == [],
    431, 434
tcp.analysis.window_full, 120, 432, 436
tcp.analysis.window_update, 436
tcp.analysis.zero_window, 120, 432, 436
tcp.checksum.status == "Bad", 432, 436
tcp.checksum_bad == 1, 114
tcp.dstport == 443, 105
tcp.flags.fin == 1, 120
tcp.flags.reset == 1, 73, 113, 179
tcp.flags.syn == 1 && tcp.flags.ack == 0, 73, 120
tcp.flags.syn == 1 && tcp.flags.ack == 0 &&
    tcp.window_size <= 1460, 480
tcp.len > 0, 74, 121
tcp.options.sack_le, 432
tcp.port == 443 && ip.addr == <>, 117
tcp.stream == [], 431, 425, 434
tcp.stream eq 0, 42
tcp.time_delta > 0.5, 114
telnet, 317, 322
tftp, 332
tls, 94, 98, 114
tls && tcp.port == 5061, 414
tls || tcp.port == 443, 116
tls.alert_message, 121

```

- tls.handshake, 98
- tls.handshake.certificate, 103, 117
- tls.handshake.ciphersuite, 102
- tls.handshake.extensions_server_name, 102, 121, 343
- tls.handshake.type, 103
- tls.handshake.type == [], 95, 96
- tls.record.content_type, 103
- tls.record.content_type == 22, 95
- tls.record.version, 102
- udp or icmp, 248
- udp.dstport == 53, 37, 179
- udp.port == 5060 || udp.port >= 10000, 117
- udp.srcport == 53, 37
- vlan, 154
- vlan and not vlan.id == 20, 92
- vlan.id, 92
- vlan.id == [], 92
- vlan.id == 10 && icmp, 153, 154
- vlan.id == 10 or vlan.id == 20, 92
- vlan.id == 20 and icmp, 92
- vlan.priority == 5, 92
- vlan.priority >= 5, 157
- vlan.priority >= 5 and ip, 92
- filtrowanie przechwyconych danych, 30, 36
- filtry
 - do analizy ruchu sieciowego, 113
 - warstwy drugiej, 153
- flaga
 - ACK, 43
 - FIN, 44
 - SYN, 43
- format
 - JSON, 107, 108
 - Maildir, 345
 - MMDB, 209
- fragmentacja pakietów, 212, 217, 221
 - w IPv4, 214, 466–469, 483
 - w IPv6, 219
 - zaciemniająca analizę, 252
- FTP
 - analiza sesji, 328
 - transfer danych, 327
 - uwierzytelnienie, 327
- funkcjonalność
 - Remote Switched Port Analyzer, 89
 - span-port, 63

G

- gadatliwa aplikacja, application chattiness, 454, 456, 458
- Generic Routing Encapsulation
 - IP, 272
 - Protocol Type, 273
 - Sequence Number, 273

- GeoLite2, 209
- geolokalizacja, 209
- gniazda, sockets, 373
- GNS3
 - dwa przełączniki warstwy drugiej, 90
 - dwa routery i routing statyczny, 260
 - redundantne połączenie przełączników, 126
 - router jako serwer DHCP, 232
 - sieć z dwoma routerami, 66
 - sieć z przełącznikiem, 64
 - sieć z usługami DNS, 295
 - stacja robocza i router, 41
 - stacje robocze, przełącznik i router, 159
 - stacje robocze, router i stacja KALI, 171
 - topologia z protokołem OSPF, 278
 - topologia z przełącznikiem warstwy drugiej, 143
 - topologia z redundantnymi łączami, 149
 - topologia z trzema przełącznikami, 175
 - topologia z VoIP, 384
 - trzy routery i protokół OSPF, 223
- graf przepływu, 61
- Graf strumienia TCP
 - Round Trip Time Graph, 60
 - Sekwencja (tcptrace), 438
 - Throughput Graph, 61
 - Time-Sequence Graph, 60
 - Window Scaling Graph, 61
- GRE, Generic Routing Encapsulation, 270, 483

H

- handshake, 95, 98
 - abbreviated, 100
 - TLS, 343
 - brak cipher suite, 337
 - Client Hello, 481
 - mechanizm SNI, 342
 - MTU, 343
 - niezgodność parametrów, 335
 - PMTUD, 343
 - Server Hello, 485
 - zgodność parametrów, 336
- hasła OSPF, 289
- Hierarchia protokołów, 484
 - kolumna Bajtów, 55
 - kolumna Bajty [%], 55
 - kolumna Bity/s, 55
 - kolumna Krańcowych bajtów, 55
 - kolumna Krańcowych bitów/s, 55
 - kolumna Krańcowych pakietów, 55
 - kolumna Pakiety, 55
 - kolumna Pakiety [%], 55
 - kolumna PDUs, 55
- HLB, Head-of-Line-Blocking, 359
- Hop Limit, 182, 483

HTTP, 418

- / Load Distribution, 421
- / Packet Counter, 419, 453
- / Requests, 420, 457, 459
- / Request Sequences, 422

HTTP2

- Average / Min Val / Max Val, 110
- Burst Rate, 110
- Burst Start, 110
- Count, 110
- Percent, 110
- Rate (ms), 110

HyperText Transfer Protocol 2

- DATA, 108
- JSON, 108
- Stream ID, 108
- videoid, 108

I

ICMP

- Destination unreachable, 246, 260, 286
- Echo, 261
- Echo request, 262
- error abuse, 246
 - błędy transportowe hosta, 246
 - spoofing adresów źródłowych, 248
 - wymuszanie ICMP Time Exceeded, 250
- reconnaissance, 242
- Time Exceeded, 250, 259, 261, 483
- Time-to-live exceeded, 261, 264

IEEE

- 802.1, 170, 174
- 802.11, 122
- 802.1AB, 154
- 802.1AX, 124, 148, 152
- 802.1D, 123, 130, 143, 174, 175
- 802.1w, 123, 130
- 802.2, 148
- 802.3, 146, 170

ikona

- Opcje przechwytywania, 13, 15, 17
- prezentująca folder, 15
- Restartuj aktualne przechwytywanie, 15
- Uruchom przechwytywanie pakietów, 15
- Wczytaj ponownie plik, 16
- Zamknij ten plik, 15
- Zapisz, 15
- Zatrzymaj przechwytywanie pakietów, 15
- Znajdź pakiet, 16

ikonki

- opisy danych w strumieniu, 112

interfejs trunk, 90, 135

Internet Protocol Version 4

- Fragment Offset, 216, 255
- Fragmented IP protocol, 255

Header Checksum, 267, 270

ICMP Echo request, 182

ICMP id, 183

ICMP sequence, 183

Identification, 183, 184, 217, 255, 269

IP Identification, 183

Protocol, 269

Source GeoIP, 210

Time to Live, 182–185

IP spoofing, 238

IPv4 Statistics, 470

ISN, Initial Sequence Number, 377

J

jitter, 483

K

karta sieciowa, NIC, 133

klient pocztowy Thunderbird, 345

klucze sesyjne TLS, 104

kolorowanie reguł, 49

kolumna

tworzenie, 45, 48, 223, 454

komenda

\$env:SSLKEYLOGFILE, 104

\$env:SSLKEYLOGFILE = \$path, 104

./Linphone-5.2.6.AppImage, 390

arp -d, 52

arp.spoof on i net.sniff on, 166

asterisk -V, 388

cd ~/chatty_http, 455

curl http, 198, 455

curl -I https://wikipedia.org, 305

DATA, 350, 352

disc del dev ens34 root 2>/dev/null, 441

EHL0, 349, 352, 357

ftp 172.30.1.20, 328

ifconfig, 84

ip addr, 401

ipconfig /displaydns, 31

ipconfig /flushdns, 32, 296

iperf3 -c, 87

iperf3 -c [] -u -b 800M -l 32 -t 20, 466

iperf3 -s, 84, 430, 437

iperf3 -s -p 9000, 206

LOGIN, 356

ls, 328

MAIL FROM, 349, 352

mkdir -p ~/chatty_http/icons, 455

mkdir -p ~/http_size_test, 461

mkdir -p ~/tsslab && cd ~/tsslab, 335

monitor session [] destination interface [], 65, 91

monitor session [] destination remote vlan [], 90

monitor session [] source [], 65

```

monitor session [] source interface [], 90
monitor session [] source remote vlan [], 91
nano ~/chatty_http/index.html, 455
nano server.py, 461
net.core.netdev_max_backlog=10, 206
net.probe on, 166
netsh interface ipv6 delete destinationcache,
221
nmap -p 143,993 [], 353
nmap -p 25,465,587 [], 356
no spanning-tree [], 127
nslookup, 32
nslookup -type=SRV wikipedia.org, 303
nslookup wikipedia.org 8.8.8.8, 304
nslookup wikipedia.org, 296–299
OPTS UTF8 ON, 329
PASS, 328, 329, 333
ping, 64, 66
ping -c, 468
ping helion.pl, 32
ping -M, 469
PORT, 329, 330
python3 ~/http_size_test/server.py, 461
python3 -m http.server 8000, 455
python3 -m http.server 8080, 192, 198
QUIT, 331
RCPT TO, 349, 352, 357
remote-span, 90
resolvectl flush-caches, 296
RETR test.txt, 331
router-id [x.x.x.x], 281
set arp.spoof.internal true, 166
set arp.spoof.targets [], 166
show ip ospf neighbor, 279
show spanning-tree, 127, 175, 176
sip show peers, 389, 393, 396
STARTTLS, 348
sudo adduser ftpuser, 328
sudo adduser user1, 346
sudo apt install dnsmasq, 295
sudo apt install vsftpd, 328
sudo apt install -y asterisk, 388
sudo apt install -y postfix dovecot-imapd
mailutils, 345
sudo apt update, 295, 328, 345, 388
sudo apt update && sudo apt install -y
isc-dhcp-server, 313
sudo apt -y upgrade, 345
sudo arp-scan - - localnet, 159
sudo asterisk -rvvv, 393
sudo bettercap -iface [], 166
sudo doveconf -n | grep mail_location, 347
sudo ethtool -K ens34 gro off gso off tso off,
437, 448
sudo ip link set dev ens34 mtu 600, 468
sudo ip route add default via [], 451
sudo iptables -A OUTPUT -p udp --sport 53 -j
DROP, 299
sudo iptables -D OUTPUT -p udp --sport 53 -j
DROP, 300
sudo kill -CONT <PID>, 207
sudo kill -STOP <PID>, 207
sudo nano /etc/asterisk/sip.conf, 388, 393, 396
sudo nano /etc/dnsmasq.conf, 295
sudo nano /etc/dovecot/conf.d/10-mail.conf,
346
sudo nano /etc/dovecot/conf.d/
↳10-namespaces.conf, 346
sudo nano /etc/dovecot/dovecot.conf, 346
sudo nano /etc/postfix/main.cf, 345
sudo nmap -sS 192.168.1.0/24, 161
sudo passwd adam1, 346
sudo pkill dhcpcd, 313
sudo python3 frag_timeout.py, 257
sudo python3 -m http.server 80, 351
sudo sed -i '/203\,0\,113\,10
wikipedia\.org/d' /etc/hosts, 307
sudo sh -c 'echo [] >> /etc/hosts', 305
sudo sysctl -w net.ipv4.ip_forward=1, 451
sudo systemctl enable postfix, 345
sudo systemctl restart asterisk, 389, 393, 396
sudo systemctl restart dnsmasq, 295, 297–300,
305, 307
sudo systemctl restart dovecot, 346
sudo systemctl restart isc-dhcp-server, 313
sudo systemctl restart postfix, 345
sudo systemctl status isc-dhcp-server, 313
sudo systemctl stop dhcpcd, 313
sudo systemctl stop isc-dhcp-server, 313
sudo tc qdisc, 80
sudo tc qdisc add dev [] root netem delay []
distribution normal, 437
sudo tc qdisc add dev ens33 root netem delay []
distribution normal, 401
sudo tc qdisc add dev ens34 root netem delay [],
452, 458
sudo tc qdisc add dev eth0 root netem delay [],
297
sudo tc qdisc change dev ens33 root netem
delay [] distribution normal loss [], 402
sudo tc qdisc change dev ens34 root netem
delay [] distribution normal loss [], 437
sudo tc qdisc del dev [] root 2>/dev/null, 437
sudo tc qdisc del dev [] root, 84, 203
sudo tc qdisc del dev ens34 root 2>/dev/null,
438, 448, 449, 458
sudo tc qdisc del dev eth0 root, 299
sudo tc qdisc show dev ens33, 401
sudo tcpdump -i eth0 -n port 67 or port 68, 314
sudo yersinia -I, 175
switchport mode trunk, 135
switchport trunk encapsulation dot1q, 135

```

komenda

- sysctl -w net.ipv4.ip_forward=1, 192
- systemctl status asterisk, 388
- systemd-resolve --flush-caches, 296
- traceroute, 184, 226, 486
- USER, 328, 329
- watch -n 1 "dhcp-lease-list", 314

komunikacja

- ARP, 51, 67
- DNS, 36, 295
- FTP, 330
- ICMP, 36, 71
- pocztowa, 348
- pocztowa zaszyfrowana, 349
- RTP, 387, 392
- TCP, 42
- TFTP, 331
- TLS, 337
- VoIP, 386
- w DNS, 30
- z routerami, 65

- komunikat ICMP Destination unreachable, 70

- komunikaty TCP Dup ACK, 81

Konfiguracja profili

- pozycja Profil: Default, 45

konfiguracja

- funkcjonalności span-port, 63
- przełącznika, 63

- kontrola przepływu, flow control, 42

L

- LACP, Link Aggregation Control Protocol, 148, 483

Link Aggregation Control Protocol

- Actor, 151
- Actor Information, 151
- Actor Key, 152
- Actor Port, 152
- Actor Port Priority, 152
- Actor State, 152
- Actor System Priority, 152
- Aggregation, 152
- Collecting, 152
- Collector Information, 152
- Distributing, 152
- LACP Activity, 152
- Partner, 151
- Partner Information, 151
- Partner Key, 152
- Partner Port, 152
- Partner State, 152
- Partner System, 152
- Partner System Priority, 152
- Synchronization, 152

Link Layer Discovery Protocol

- Capabilities, 141
- Chassis ID, 140
- Management Address, 141
- Port Description, 141
- Port ID, 141
- System Description, 141
- System Name, 141
- Time To Live, 141

Linphone

- instalacja aplikacji, 384
- konfiguracja aplikacji, 390
- okno główne, 385
- tworzenie konta SIP, 390

- little-endian, 69

- LLC, Logical Link Control, 146, 483

- LLDP, Link Layer Discovery Protocol, 483

Logical Link Control

- Control field, 147
- DSAP, 147
- SSAP, 147

logowanie

- automatyczne, 322
- automatyczne SSHv2, 323
- automatyczne Telnetu, 323
- charakterystyka ruchu sieciowego, 316

M

MAC

- flapping, 145
- flooding, 170

mechanizm

- Duplicate Address Detection, 189
- EtherChannel, 482
- Fast Retransmit, 205
- MSS Clamping, 276
- NONCE, 368
- parowania, 33
- PMTUD, 218
- rekonstrukcji fragmentów, 256
- Slow Protocols, 149
- SNI, 342
- tc netem, 297, 298
- three-way handshake, 40
- uRPF, 259

menu

- Edycja, 47
- główne
 - ikony, 15
- Narzędzia, 155
- Plik, 24
- podręczne
 - Filtr konwersacji, 53
 - Hierarchia protokołów, 54
 - Koloruj z filtrem, 50

- Pre-Master -Secret log filename, 108
- Przeglądaj, 108
- Strumień RTP, 401
- Transport Layer Security, 108
- Ustawienia protokołów, 108
- Utwórz kolumnę z pola, 223
- Zastosuj filtr, 39
- Pomoc, 76
- Przechwytywanie, 417
- Statystyki, 54
- Telefonia, 401
- Telephony, 117
- Widok, 50, 79
- MSS, Maximum Segment Size, 43, 212, 483
- MSTP, Multiple Spanning Tree Protocol, 483
- MTU, Maximum Transmission Unit, 212, 343, 468–470, 484

N

- NAC, Network Access Control, 312
- Narzędzia
 - Bloki adresów MAC, 155
- narzędzie
 - arp-scan, 159
 - cURL, 192
 - GNS3, 41
 - hping3, 249
 - Hydra, 355
 - I/O Graphs, 244
 - iperf, 80, 203
 - macof, 171
 - nmap, 158
 - npcap, 11
 - nping, 242
 - openssl s_client, 334
 - openssl s_server, 334
 - swaks, 357
 - tc netem, 401
 - traceroute, 486
 - yersinia, 175
- narzut opóźnienia, latency penalty, 460
- NAT, Network Address Translation, 484

O

- ocena
 - pliku przechwytywania, 415
 - wiarygodności przechwytywania, 465
- odpowiedź DNS, 35, 299
- odszukiwanie wartości strumienia, 108
- odszyfrowywanie ruchu TLS, 103, 106
- ogranicznik przepływu, bottleneck, 460
- okno
 - Analiza strumienia RTP, 408
 - Bloki adresów MAC, 156

- Conversations, 56
- Drukuj, 30
- Eksportuj wybrane pakiety, 25
- Endpoints, 212
- Expert Information, 415, 444
- Export PDUs, 27
- Filtry wyświetlania, 38
- główne
 - kolumna Destination, 14, 22, 33
 - kolumna Info, 14, 22, 23, 33
 - kolumna Length, 14, 23
 - kolumna No., 13, 22
 - kolumna Protocol, 14, 22
 - kolumna Source, 13, 22
 - kolumna Time, 13, 22
 - pole Przechwytywanie, 12
 - wybór interfejsu sieciowego, 12
 - Zarządzaj filtrami wyświetlania, 37
- Hierarchia protokołów, 54
- HTTP / Load Distribution, 421
- HTTP / Packet Counter, 419
- HTTP / Request Sequences, 422
- HTTP / Requests, 420
- HTTP2, 111
- Informacja ekspercka, 82
- IPv4 Statistics / All Addresses, 471
- IPv4 Statistics / Destinations and Ports, 472
- IPv4 Statistics / IP Protocol Types, 472
- IPv4 Statistics / Source and Destination Addresses, 474
- IPv4 Statistics / Source TTLS, 473
- klucza, 109
- Konfiguracja profili, 45
- MaxMind Database Paths, 211
- O programie Wireshark, 475
- Odtwarzacz RTP, 407
- Opcje przechwytywania, 13, 15, 18, 418
- Packet Lengths, 57
- Połączenia VoIP, 405
- Preferencje, 47
- Przepływ, 61
- Punkty krańcowe, 57
- Reguły kolorowania Default, 51
- Scal pliki przechwytywania, 29
- Szczegóły pliku przechwytywania, 63, 416
- Unsaved packets, 15
- Wykresy we./wy., 59
- Zapisz plik przechwytywania jako, 15
- Zapisz plik przechwytywania jako, 24
- Zarządzaj interfejsami, 18
- Opcje przechwytywania
 - Opcje, 21
 - Aktualizuj listę pakietów w czasie rzeczywistym, 21
 - Automatyczne przewijanie podczas przechwytywania, 21

Opcje przechwytywania

Opcje

- Pokaż informacje o przechwytywaniu podczas trwającego przechwytywania, 21
- Rozwiąż adresy MAC, 22
- Rozwiąż nazwy sieciowe, 22
- Rozwiąż nazwy warstwy transportowej, 22
- pozycja Katalog plików tymczasowych, 22
- pozycja Zatrzymaj przechwytywanie automatycznie po..., 22

Rozpocznij, 13

Rozwiąż nazwy sieciowe, 13

Wejście, 17, 417

- kolumna Filtr przechwytywania, 19
- kolumna Interfejs, 18
- kolumna Nagłówki warstwy łącza, 19
- kolumna Rozmiar przechwytywania, 19
- kolumna Rozmiar przechwyconej ramki, 417
- kolumna Ruch, 18
- kolumna Tryb mieszany, 19
- kolumna Tryb monitora, 19
- parametr snaplen, 418
- pozycja Filtr przechwytywania dla wybranych interfejsów, 19
- Wpisz filtr przechwytywania, 19
- Zarządzaj interfejsami, 18

Wyjście, 20

- opcja Stwórz nowy plik automatycznie..., 21
- Przeglądaj..., 21

Open Shortest Path First

- Area ID, 228
- Backup Designated Router, 291
- DB Description, 281, 285–287
- Dead Interval, 280, 281
- Designated Router, 291
- Hello, 279
- Hello Interval, 280, 281, 291
- Hello Packet, 227, 280, 281, 291
- LS Age, 279
- LS Request, 281
- LS Update, 229, 279, 281, 285
- LS Update Packet, 279
- Network Mask, 291
- Number of Links, 279
- Router Dead Interval, 291
- Router-LSA, 279

opóźnienie

- DNS, 297
- w warstwie trzeciej, 196

OUI, Organizationally Unique Identifier, 147, 484

out-of-order, 484

P

PAgP, Port Aggregation Protocol, 148

pakiety

- ACK, 87
- Application Data, 98
- Client Hello, 96–100, 366
- DHCP, 232
- DNS, 36
- handshake'u TLS, 95
- HTTP2, 106, 107
- ICMP, 20, 36, 68, 215, 223, 225
- ICMP Echo reply, 237, 240
- ICMP Echo request, 237, 239, 240
- ICMP reply, 28, 29
- ICMP request, 28, 29
- IP, 181
- LS Update, 229
- OSPF, 70
- Server Hello, 97, 101, 368
- SYN, 274
- TCP out-of-order, 85
- TCP Retransmission, 81
- UDP, 164, 225

pasywny sniffing, 289

PDU, Protocol Data Unit, 26

pełny przebieg, full handshake, 97

PFS, Perfect Forward Secrecy, 97

Plik

- Drukuj, 30
- Eksportuj bajty pakietu, 26
- Eksportuj klucze sesji TLS, 28
- Eksportuj PDU do pliku, 26
- Eksportuj prezentację pakietów, 25
- Eksportuj wybrane pakiety, 24
- Ostatnio otwarte, 23
- Otwórz, 23
- Scal..., 29
- Usuń nagłówki, 27
- Zapisz plik przechwytywania jako, 23

plik

- extensions.conf, 389
- rtp.conf, 389
- z ICMP request, 28

pliki

- .csv, 26
- .lua, 474
- .pcap, 376
- .pcapng, 28
- .pdf, 30
- .png, 26
- przechwytywania, 415, 467
- PMTUD, Path MTU Discovery, 218, 344, 466, 468, 484

- poczta elektroniczna, 345
 - analiza sesji, 345
 - atak na IMAPS, 353
 - atak na SMTP, 356
 - atak phishingowy, 351
 - próbna odgadnięcia hasła, 355
- podejście do analizy
 - faza inicjalizacji TLS, 77
 - transfer zaszyfrowanych danych, 78
 - transmisja danych od klienta, 78
- polecenia powłoki PowerShell, 360
- polecenie, *Patrz* komenda
- połączenie trunk, 134
- Pomoc
 - Sprawdź aktualizacje..., 76
- porty
 - rola, 125
 - stan, 125
- Postfix, 345
- Preferencje
 - (Pre)-Master-Secret log filename, 106
 - MaxMind Database Directory, 210
 - Message Authentication Code, 110
 - NameResolution, 210
 - Pre-Shared Key, 110
 - Przeglądaj, 106
 - RSA keys list, 109
 - TCP
 - opcje, 196
 - TLS, 109
 - UDP, 370
- profile, 44
- program, *Patrz* narzędzie
- protokoły
 - hierarchia, 54
 - komunikacyjne, 369
 - statystyki, 54
 - tekstowe, 326
 - transportowe, 59
- protokół
 - ARP, 51, 165, 189
 - CDP, 137
 - DHCP, 127, 231, 311
 - DNS, 33, 310
 - FTP, 326
 - HTTP/2, 103
 - HTTPS, 104
 - ICMP, 67, 241
 - IMAP, 345
 - IP, 238
 - IPv6, 219
 - LACP, 148
 - LLDP, 140
 - OSPF, 70, 222, 227, 278
 - POP3, 351
 - QUIC, 55, 358
 - RTCP, 383
 - RTP, 382
 - SDP, 382
 - SIP, 382, 411
 - SMTP, 345
 - SSH, 319
 - STP, 122
 - TCP, 40, 43
 - Telnet, 317
 - TFTP, 326
 - TLS, 93
 - UDP, 358
- Przechwytywanie
 - Opcje, 417
- przechwytywanie
 - komunikacji DNS, 32
 - rozmów VoIP, 385
 - ruchu ICMP, 17
- przełącznik
 - konfiguracja, 63
 - połączenie trunk, 134
 - tablica MAC, 143
- przepełnienie tablicy CAM, 170
- Przepływ
 - Ogranicz do filtru wyświetlania, 62
 - Typ przepływu, 62
- przycisk
 - filtrowania, 41
 - filtru wyświetlania, 40
 - Otwórz, 29
 - Przeglądaj..., 21
 - Zachowaj, 23, 25
 - Zarządzaj interfejsami, 18
- punkty krańcowe, endpoints, 57, 461

Q

- QoS, 299
- QUIC, Quick UDP Internet Connection, 358, 484
 - analiza protokołu, 358
 - bez klucza, 361
 - Client Hello, 366
 - długi nagłówek, 362
 - krótki nagłówek, 365
 - limity, 368
 - odszukiwanie transmisji, 361
 - pakiety uzgodnienia, 368
 - parametry rozszerzeń, 367
 - parametry transportowe, 367, 368
 - ruch po rozszyfrowaniu, 364
 - Server Hello, 368

R

ramka

- ARP, 145, 160
- BPDU, 130, 131
- CDP, 138, 150
- DTP, 147
- ethernetowa, 134
- JSON, 107
- LACP, 150, 151
- LLC, 146
- LLDP, 141
- SNAP, 146
- sterująca, 137
- STP, 132, 150
- unicast, 145

RCODE, 484

rekonesans, 158, 241

Resolver, 484

RFC, 9

RFC 1034, 295–299, 306, 308

RFC 1035, 30, 34, 295, 297–299, 301, 303, 305, 308, 309

RFC 1042, 71

RFC 1122, 189, 246, 247, 374, 381

RFC 1123, 34

RFC 1191, 425

RFC 1350, 326, 331–333

RFC 1812, 182, 185, 259, 260, 264, 470, 473

RFC 2003, 265, 267, 269

RFC 2018, 432, 442

RFC 2131, 233, 311, 314, 315

RFC 2132, 316

RFC 2177, 355

RFC 2308, 301–303, 306

RFC 2326, 405

RFC 2328, 231, 278, 281–288, 291, 292

RFC 2460, 371, 372

RFC 2616, 451

RFC 2782, 31, 303

RFC 2784, 265, 266, 270, 276, 277

RFC 2890, 265, 270

RFC 3118, 311

RFC 3261, 382, 385, 387, 391, 393, 396–398, 404, 405, 410, 411

RFC 3264, 398, 399

RFC 3501, 348

RFC 3550, 117, 382, 383, 386, 387, 392–394, 396, 397, 402, 403, 406–408, 411

RFC 3551, 399, 401, 406

RFC 3596, 31

RFC 3611, 383, 387, 411

RFC 3711, 383, 411

RFC 4033, 31, 306

RFC 4034, 31, 306

RFC 4035, 31, 33, 34, 306

RFC 4217, 333

RFC 4251, 319, 333

RFC 4252, 333

RFC 4253, 319, 321, 333

RFC 4254, 333

RFC 4301, 269

RFC 4443, 185

RFC 4566, 382, 385, 399

RFC 4733, 399

RFC 4861, 189, 190

RFC 4862, 189, 191

RFC 4960, 405

RFC 5155, 31

RFC 5227, 186, 187, 189

RFC 5246, 100, 114, 335–341, 348

RFC 5321, 349, 352, 356, 357

RFC 5322, 350

RFC 5340, 278

RFC 5681, 197, 200, 202–205, 207, 208

RFC 6066, 342

RFC 6125, 341–343

RFC 6265, 460, 462, 463

RFC 6298, 197–200, 424, 429, 432, 436, 438, 440, 441, 445, 449

RFC 6698, 31

RFC 6840, 33, 34

RFC 6864, 183, 293

RFC 6891, 34

RFC 6960, 420, 422

RFC 6994, 380

RFC 7323, 379, 448

RFC 7540, 359

RFC 768, 371, 385, 392, 424, 470

RFC 7826, 405

RFC 791, 182, 183, 216, 256–259, 267–269, 276, 423, 426, 466, 467, 469, 470, 472

RFC 792, 218, 387, 392, 466, 467, 469

RFC 793, 113, 197–200, 203–207, 344, 347, 359, 374, 375, 377, 381

RFC 7932, 465

RFC 815, 253, 256–258

RFC 8200, 371, 423, 426, 466, 467

RFC 8201, 425, 466, 468

RFC 826, 165, 186, 187, 324, 354, 385

RFC 8314, 348, 349, 353, 355

RFC 8446, 100, 114, 116–118, 335, 336, 340, 344, 348, 355, 361

RFC 8489, 386

RFC 854, 317, 326

RFC 8825, 383

RFC 9000, 358–360, 362, 365, 383

RFC 9001, 358, 361, 365, 368

RFC 9002, 358

RFC 9110, 114, 359, 419–422, 456, 458, 460, 462

RFC 9112, 419, 420, 422

RFC 9113, 359
 RFC 9114, 358, 359, 365
 RFC 9293, 113, 423–432, 436, 439–445, 448, 449, 470
 RFC 9308, 358
 RFC 9312, 359
 RFC 9369, 359
 RFC 9460, 307
 RFC 959, 326–331
 router, 65, 69, 192
 routing, 222, 227

- błędne czasy Hello i Dead Interval, 279
- błędne parametry interfejsu, 282
- konflikt Router-ID, 281

 rozgłoszenie ARP, 52
 rozmiar

- okna, 43
- segmentu MSS, 43

 RSPAN, Remote Switched Port Analyzer, 89

- przechwytywanie ruchu, 89

 RST, 484
 RSTP, Rapid Spanning Tree Protocol, 484
 RTCP, Real-Time Control Protocol, 485
 RTO, Retransmission Timeout, 199, 485
 RTP, Real-Time Transport Protocol, 382, 485

- analiza strumienia, 117, 406, 409
- czasy pakietów, 402
- dane statystyczne połączenia, 403
- identyfikacja strumieni, 398
- odtworacz, 407
- statystyki strumienia, 401

 RTT, Round-Trip Time, 58, 69, 199, 379
 ruch multimedialny, 381

S

Scal pliki przechwytywania

- Otwórz, 29

 SDP, Session Description Protocol, 382, 400, 485

- w SIP, 398

 Server Hello, 97, 101, 368, 485
 serwer

- DNS, 31
- pocztowy Dovecot, 347
- pocztowy Postfix, 345

 sieci VLAN, 133
 Simple Mail Transfer Protocol

- Content-Type, 350
- DATA, 350
- Date, 350
- From, 350
- Message-ID, 350
- Subject, 350
- To, 350
- User-Agent, 350

 SIP, Session Initiation Protocol, 382, 387, 485

- centrala jako B2BUA, 393
- centrala jako B2BUA z transkodowaniem, 395
- konfiguracja centrali, 388
- realizacja połączenia, 399
- statystyki, 409
- transmisja, 391
- tworzenie konta, 390
- User Agent Client, UAC, 382
- User Agent Server, UAS, 382
- ustanawianie połączenia, 397

 skanowanie

- sieci, 158
- sieci za pomocą nmap, 158
- TCP SYN, 160
- UDP, 162

 skrócone wznowienie sesji, session resumption, 97
 skrócony uścisk dłoni, abbreviated handshake, 98
 skrót klawiaturowy, 17
 skrypty Lua, 474
 SNAP, Subnetwork Access Protocol, 147, 485
 SNI, Server Name Indication, 94, 100, 342, 367, 485
 sniffing, 289
 SPAN

- przechwytywanie ruchu, 91

 Spanning Tree Protocol

- BPDU Type: Configuration (0x00), 130
- Bridge Identifier, 131, 132, 178
- Forward Delay, 131, 132
- Hello Time, 131
- Max Age, 131
- Message Age, 131
- Protocol Identifier, 130
- Protocol Version Identifier, 130
- Root Identifier, 131, 132, 177
- Root Path Cost, 132, 178
- Topology Change, 131, 132, 178
- Topology Change Acknowledgment, 131

 span-port, 63
 SRTP, Secure Real-Time Transport Protocol, 485
 SSAP, Source Service Access Point, 147, 485
 SSH, 319

- błędne uwierzytelnienie, 321
- logowanie automatyczne, 323

 Statystyki

- Conversations, 128, 148
- Długości pakietu, 57
- Graf przepływu, 61
- Graf strumienia TCP, 438
- Hierarchia protokołów, 54, 148, 276
- HTTP, 419, 452, 457
- HTTP2, 105, 110
- IPv4 Statistics, 470
- Punkty końcowe, 148, 210
- Szczegóły pliku przechwytywania, 62, 415, 465
- Wykresy we./wy., 58, 128

sterowanie przepływem i przeciążeniem, 205
 STP, Spanning Tree Protocol, 122, 485
 algorytm działania, 124
 analiza, 130
 manipulation, 174
 opis, 122
 wyznaczanie ról i stanów portów, 125
 strumienie TCP, 425
 studium incydentu, 284–288
 symulacja
 ograniczenia przepustowości, 86
 ruchu TCP, 192
 SYN, SYNchronization, 40
 SYN-ACK, 40
 Szczegóły pliku przechwytywania, 62, 416, 467
 szyfrowanie
 sesji, 333
 transmisji TCP, 77

T

tabela połączeń, 201
 tablica
 ARP, 66, 70
 CAM, 170
 DNS, 31
 MAC, 143
 routingu, 69, 72, 284
 TCP, Transmission Control Protocol, 328–331, 373
 Allow subdissector to reassemble TCP
 streams, 375
 analiza kierunkowa strumieni, 428
 Analyze TCP sequence numbers, 376
 Calculate stream packet number and
 timestamps, 377, 378
 Client port dissectors/TCP UDP port(s), 380
 Display process information via IPFIX, 380
 Do not call subdissectors for error packets, 380
 Dup ACK, 194, 195
 Duplicate ACK, 435
 Evaluate bytes in flight based on sequence
 numbers, 377, 378
 Fast Retransmission supersedes Out-of-Order
 interpretation, 379
 Ignore TCP Timestamps in summary, 379
 nagły spadek przepustowości, 429
 niezgodność nazwy hosta, 341
 okresowe fluktuacje, 433
 out-of-order, 207
 Read the seq no. as syn cookie, 380
 Reassemble out-of-order segments, 375
 Relative sequence numbers, 377
 Retransmission, 193, 195, 207
 rozpoczęcie sesji, 44
 Show TCP summary in protocol tree, 374
 Stream Graph, 485

TCP Experimental Options using the format
 of RFC 6994, 380
 TCP UDP port(s), 381
 Track number of bytes in flight, 377
 Try heuristic sub-dissectors first, 379
 Validate the TCP checksum if possible, 374
 zakończenie sesji, 44
 Telefonnia, 404
 Analiza strumienia IAX2, 405
 ANSI, 405
 GSM, 405
 H.225, 405
 Komunikaty ISUP, 405
 Komunikaty UCP, 405
 MTP3, 405
 Operacje SMPP, 405
 Osmux, 405
 Połączenia VoIP, 404
 Przepływy SIP, 405
 RTP, 405
 RTSP, 405
 SCTP, 405
 SIP Statistics, 405
 WAP-WSP Packet Counter, 405
 Telnet, 317
 błędne uwierzytelnienie, 320
 logowanie automatyczne, 323
 prawidłowa transmisja, 325
 przejęcie sesji, 324
 TFTP
 analiza komunikacji, 331
 niezabezpieczona komunikacja, 332
 three-way handshake, 40, 42
 Thunderbird, 345
 nieprawidłowe działanie, 347
 niezaszyfrowana komunikacja, 347
 TLS, Transport Layer Security, 94, 486
 błąd negocjacji, 335
 błąd walidacji certyfikatu, 338, 339
 diagnostyka problemów, *Patrz także*
 handshake TLS
 prawidłowa komunikacja, 337
 TLV, Type-Length-Value, 138, 140
 traceroute, 184, 226, 486
 translacja NAT, 237
 transmisja
 HTTP, 48
 TCP, 88
 TLS, 94
 Transport Layer Security
 application_layer_protocol_negotiation, 100
 Certificate, 96, 98, 102
 Change Cipher Spec, 97, 102
 Cipher Suite, 101
 Cipher Suites, 99
 Client Hello, 96, 99

- Client Key Exchange, 97, 102
- ec_point_formats, 100, 101
- Encrypted Handshake Message, 97
- New Session Ticket, 97, 101
- pakiety handshake'u, 95, 97
- ramki TLS, 94
- renegotiation_info, 101
- Server Hello, 96, 98, 101
- Server Hello Done, 96–98, 102
- Server Key Exchange, 96, 98, 102
- Session ID, 99
- session_ticket, 101
- signature_algorithms, 100
- supported_groups, 100
- Trivial File Transfer Protocol
 - ACK, 332
 - DATA, 332
 - Read Request, 332
- TTL, Time To Live, 68, 141, 182–186, 222, 486
- manipulation, 259
- tunel
 - GRE, 270
 - IP-in-IP, 266
- tunelowanie, 265
- tworzenie
 - filtru, 39
 - interfejsu trunk, 90
 - kolumny, 45, 48, 223, 454
 - przycisku filtrowania, 41
 - reguły kolorowania, 50
 - tunelu GRE, 270
 - tunelu IP-in-IP, 266

U

- UDP, User Datagram Protocol, 369
 - Calculate stream packet number and timestamps, 373
 - Checksum, 371
 - Collect process flow information, 373
 - Ignore zero-value UDP checksums over IPv6, 372
 - Show UDP summary in protocol tree, 370
 - Try heuristic sub-dissectors first, 370
 - Validate the UDP checksum if possible, 371
- unicast flooding, 145
- Unsaved packets
 - Anuluj, 15
 - Kontynuuj bez zapisywania, 15
 - Zapisz przed kontynuowaniem, 15
- usługa DNS, 30
- ustawienia
 - formatu czasu, 79
 - programu, 76

- uwierzytelnianie
 - błędne
 - SSH, 321
 - Telnet, 320
 - FTP, 327
 - kryptograficzne, 333
 - wzorce zachowań, 316

V

- Virtual LAN
 - DEI: Ineligible, 136
 - ID: 10, 136
 - Priority: Best Effort (default), 136
- VLAN, Virtual LAN, 486
- VoIP, 382
 - aplikacja Linphone, 384
 - architektura, 382
 - diagram przepływu, 406
 - połączenia, 405
 - połączenie z centralą SIP, 388
 - symulacja degradacji jakości, 400
 - w środowisku szyfrowanym, 410

W

- warstwa druga modelu OSI, 122
 - adresowanie MAC, 142
 - bezpieczeństwo, 158
 - filtry, 153
 - najważniejsze problemy, 122
 - problem burzy rozgłoszeniowej, 126
 - protokół
 - CDP, 137
 - LLDP, 140
 - STP, 122
 - ramki sterujące, 137
 - sieć VLAN, 133
 - zestawienie połączeń, 148
- warstwa trzecia modelu OSI, 181, 191
 - bezpieczeństwo, 235
 - działanie routera, 192
 - geolokalizacja, 209
 - kolejność dostarczania pakietów, 202
 - opóźnienie, 196
 - problemy routingu, 227
 - protokół
 - DHCP, 231
 - ICMP, 241
 - IP, 181, 238
 - OSPF, 222
 - wykrywanie duplikatów
 - adresów ipv4, 186
 - adresów ipv6, 189
 - z pozycji TCP, 191

Widok

- Format czasu, 79
- Name Resolution/Resolve MAC addresses, 292
- Reguły kolorowania, 50
- wstrzyknięcie zapisu do tablicy routingu, 284
- wykres
 - Czas podróży (RTT Graph), 439
 - Numery sekwencyjne (Stevens), 443
 - Okno skalowania, 447, 450
 - Przepustowość, 446, 449
 - Sekwencja (tcptrace), 438, 442
- Wykresy we./wy., 58
 - aktywność ICMP, 244
 - analiza czasowa, 248
 - asymetria wolumenowa ruchu, 423
 - dekompozycja ruchu hosta, 426
 - hiperaktywność tunelu, 277
 - Interwał, 425
 - okresowe fluktuacje przepustowości, 434
 - ramki rozgłoszeniowe, 129
 - spadek przepustowości, 431
 - wykres porównawczy, 60
- wykrywanie
 - anomalii, 82, 274
 - manipulacji STP, 174
- wyodrębnianie
 - czasu aplikacji, 454
 - czasu sieciowego, 454
- wznowienie sesji, abbreviated handshake, 98

Z

- zakończenie sesji TCP, 44
- zapisywanie projektu, 23
- zapytania
 - ARP, 52, 142
 - DNS, 296
 - IPv4, 298
 - IPv6, 298
- Zero Window, 486
- zestaw szyfrów, cipher suite, 102
- zjawisko
 - gadatliwej aplikacji, 454
 - out-of-order, 205
 - segmentacji, 42
- zmienna środowiskowa
 - %LOCALAPPDATA%, 104
 - %TEMP%, 105

Ż

- żądanie
 - ARP, ARP request, 68
 - GET, 456

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 

WIRESHARK

Analiza ruchu sieciowego i wykrywanie włamań

By wiedzieć, co w sieci piszczy

Świadomość tego, co i w jaki sposób jest przekazywane za pomocą sieci, to istotna sprawa. Szczególnie w kontekście potencjalnych ataków i włamań. Jednym z programów pozwalających przechwycić, przeanalizować w czasie rzeczywistym i szczegółowo przebadać pakiety danych przesyłanych w sieciach komputerowych jest Wireshark. Umożliwia on między innymi diagnozowanie problemów sieciowych, wykrywanie ataków i optymalizowanie wydajności sieci.

Jeśli chcesz wiedzieć o nim więcej, poznać pełny pakiet możliwości Wiresharka i nauczyć się z nich korzystać, najnowsza książka Adama Józefioka z pewnością Ci w tym pomoże. Za jej sprawą między innymi opanujesz podstawowe i zaawansowane funkcje programu. Dowiesz się, jak z udziałem Wiresharka przechwytywać i analizować ruch w sieci, wykrywać włamania do niej, a także monitorować jej bezpieczeństwo. Dzięki lekturze poradnika i przetestowaniu opisanych w nim narzędzi poznasz również statystyczne i analityczne funkcje tego przydatnego oprogramowania.

Adam Józefiok

Ukończył studia doktoranckie na Politechnice Śląskiej w Gliwicach, na Wydziale Automatyki, Elektroniki i Informatyki. Specjalizuje się w tematyce sieci komputerowych (przełączanie, routing, bezpieczeństwo i projektowanie). Jest autorem publikacji polskich i zagranicznych z tej dziedziny. Brał udział w konferencjach naukowych (krajowych i międzynarodowych) dotyczących sieci komputerowych. Jest pracownikiem naukowym Katedry Sieci i Systemów Komputerowych Politechniki Śląskiej. Na co dzień administruje również bezpieczeństwem urządzeń sieciowych, konfiguruje między innymi urządzenia sieciowe (Cisco, HP), utrzymuje sieci WAN. Przez siedem lat kierował komórką realizacji wsparcia usług IT. Ma wieloletnie doświadczenie w pracy jako administrator sieci i projektant sieci komputerowych. Przez lata projektował serwerownie i tworzył projekty okablowania. Opracowywał procedury i dokumentacje projektowe. Na koncie ma wiele zrealizowanych projektów w zakresie zakupu sprzętu IT wraz z prowadzeniem procedur przetargowych, wdrożeniem, wykonaniem dokumentacji i testami. Posiada certyfikaty: CCNA Security, CCNP Routing and Switching, Cisco CCDP, CCNAV, jak również certyfikat instruktorski Cisco CCAI, certyfikaty ITIL i PRINCE2. Jego pasją jest pisanie książek, praca ze studentami i szeroko rozumiana dydaktyka.

Helion 



helion.pl



HELION S.A.
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 96 63
helion@helion.pl

KOD KORZYŚCI
Sięgnij po więcej! ▶



ISBN 978-83-289-2639-4



9 788328 926394

Cena: 149,00 zł