Przegląd wybranych programów do sniffowania

Programy do analizowania ruchu w sieciach – tzw. sniffery – są darmowe i powszechnie dostępne w sieci Internet. Do najpopularniejszych należą:

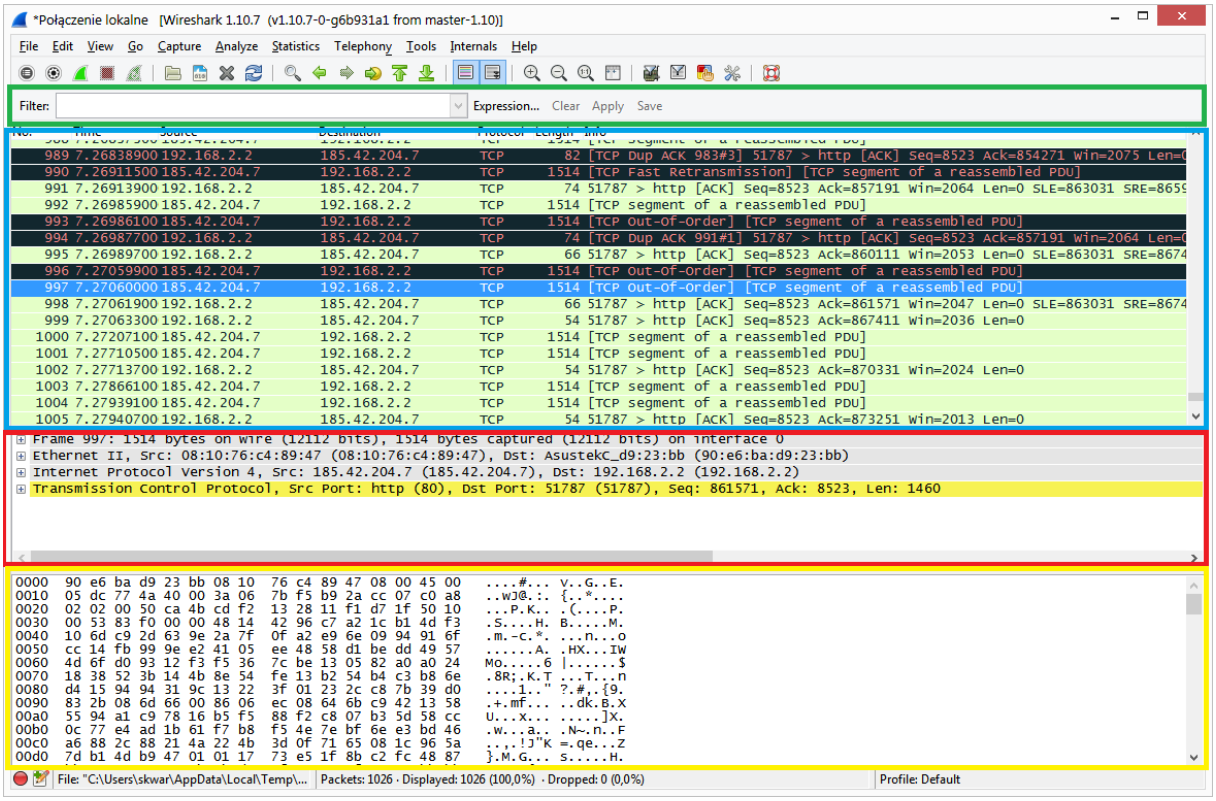
- tcpdump – udostępnia przechwytywanie i zapisywanie pakietów;
- Ettercap – potężny sniffer do przechwytywania i analizowania ruchu oraz do wykonywania zaawansowanych ataków;
- Wireshark – zaawansowane narzędzie do przechwytywania i zapisywania danych krążących po sieci;
- Snort – wykorzystywany przez administratorów sieci; posiada szeroki zakres mechanizmów detekcji włamań;
- dsniff – pakiet zaawansowanych narzędzi do przechwytywania danych i wykonywania ataków

Wszystkie wymienione sniffery korzystają z API pcap (packet capture). Dla systemów Unix-owych implementujemy je z biblioteką libpcap, dla Windows stosujemy port pcap znany jako WinPcap.

1. Wireshark

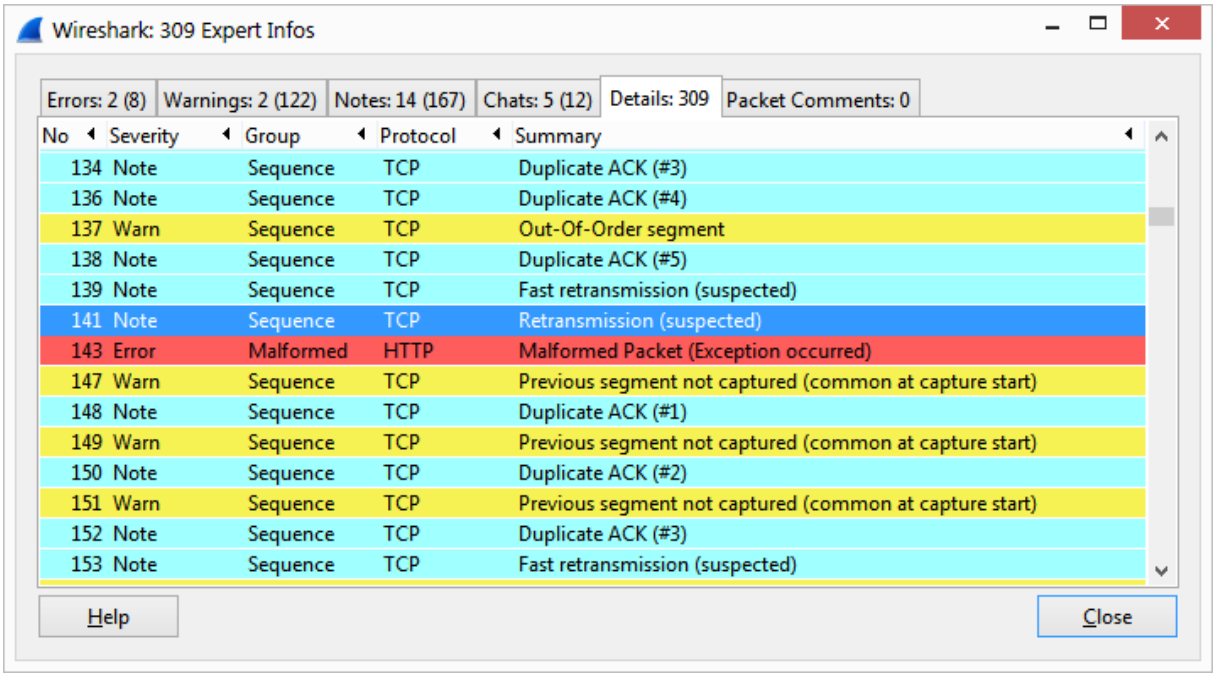
Wireshark (dawniej znany jako ethereal) potrafi dekodować wiele protokołów, przy czym rozpoznaje i śledzi strumienie także protokołów szyfrowanych, takich jak SSL. Szczególnie mocnym narzędziem są filtry - w oprogramowanie wbudowano ponad 105 tys. filtrów, obejmując najważniejsze protokoły sieciowe, dodatkowe warunki oraz inne opcje. Filtry te można łączyć z typowymi warunkami, takimi jak IP, MAC czy inne opcje.

Główne okno po rozpoczęciu przechwytywania.



- Filtry przechwytywania - Filtry używane są do wyszukiwania wewnątrz zrzuconych logów.
- Panel listy pakietów - Lista pakietów wyświetla wszystkie zrzucone pakiety. Otrzymujemy tam informacje o adresie źródłowym docelowym MAC/IP, numery portów TCP/UDP, zawartości pakietów i protokołów.
- Szczegóły pakietu - W panelu szczegółów pakietów widzimy informacje na temat zaznaczonych pakietów na liście pakietów.
- Podgląd pakietu - Panel podglądu pakietu zwany "panelem rozmiaru pakietu", przedstawia te same informacje co panel szczegółów pakietów jednak w wartościach hexadecymalnych.

Aby ułatwić diagnostykę sieci, oprogramowanie oferuje narzędzie zwane Expert Info, które sugeruje potencjalne problemy, wykryte w przechwyconym materiale. Nie należy ich traktować jako wyrocznię, ale raczej jako wskazówkę, ujawniającą miejsce dalszych poszukiwań problemów.



Ważną cechą Wireshark'a jest możliwość użycia wirtualnego interfejsu "any", który zbiera ruch ze wszystkich dostępnych interfejsów sieciowych. Ma to głęboki sens, gdyż w przypadku analizy ruchu odbywającego się w pełnym dupleksie, należy mieć dwa interfejsy analizujące, podłączone do dwóch wyjść. Ponadto w ten sposób można koncentrować ruch z dwóch segmentów sieci, zatem analizator może zbierać informacje z różnych segmentów w tym samym czasie, by dokonać korelacji zdarzeń.

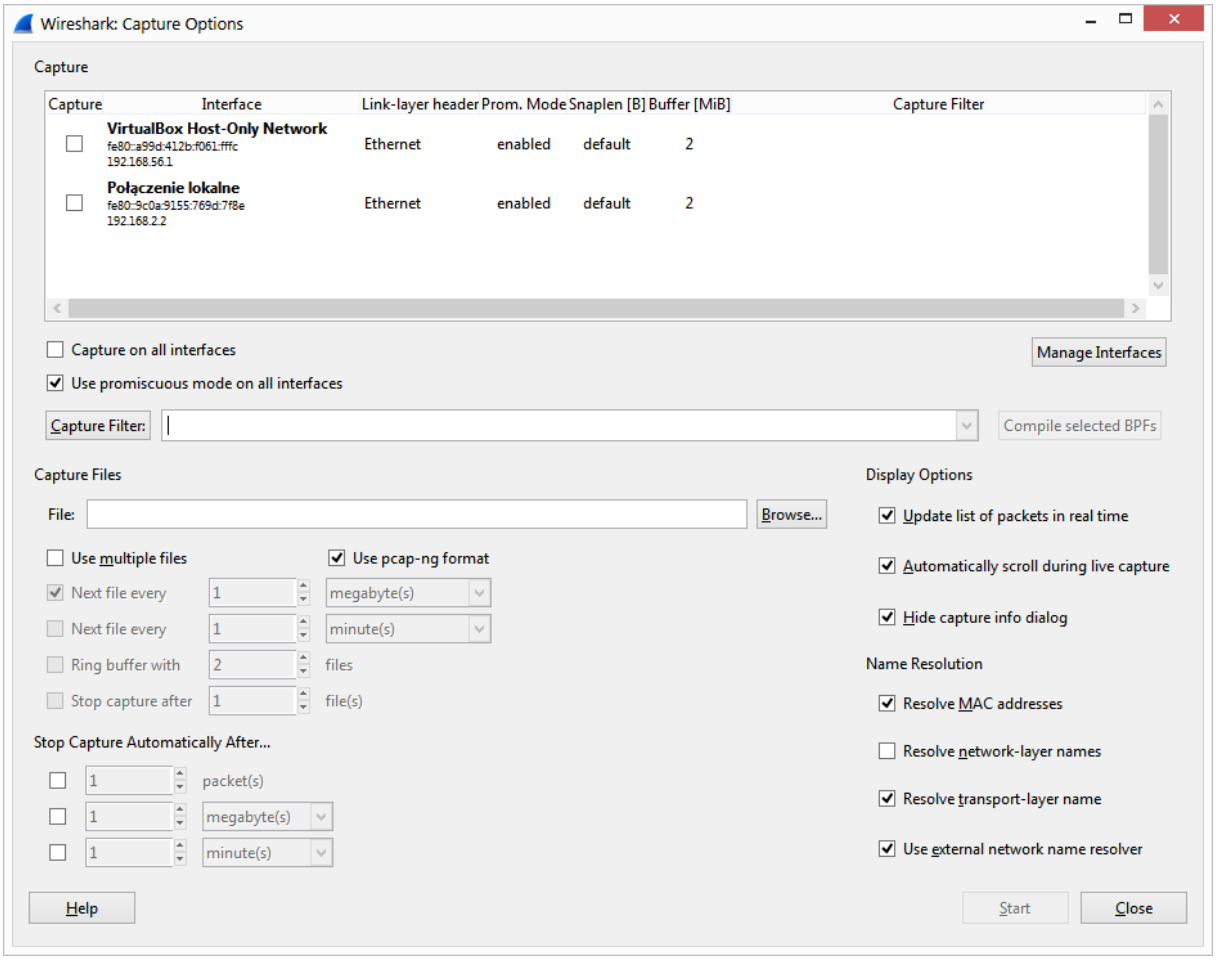
Bardzo częstym problemem podczas uruchamiania Wireshark z jego ustawieniami domyślnymi jest wyświetlana zbyt duża liczba informacji na ekranie i trudno jest wówczas odnaleźć potrzebne nam informacje.

Dlatego filtry są tak ważne. Pomagają nam wyświetlać informację, które nas interesują.

- **Filtry przechwytywania**: Używane do wyboru zapisywanych w logach.
  Są definiowane zanim rozpocznie się przechwytywanie.
- **Filtry wyświetlania**: Używane do wyszukiwania informacji wewnątrz przechwyconych już danych. Mogą być modyfikowane kiedy informacje zostały już przechwycone.

Przeznaczenie dwóch filtrów jest różne.
Filtry przechwytywania używane są jako filtry minimalizujące rozmiar przechwyconych danych, zapobiegając tworzeniu za dużych plików logów.
Filtry wyświetlania są mocniejsze (kompleksowe) i pozwalają wyszukiwać dokładnie tych informacji (danych), które są przez nas pożądane.

Filtry przechwytywania muszą być zdefiniowane zanim przechwytywanie zostanie rozpoczęte. W przypadku filtrów wyświetlana jest inaczej - one mogą być zmieniane później i w każdym momencie podczas zrzucania.



Składnia dla Filtru przechwytywania:

Składnia	Protocol	Direction	Host(s)	Value	Logical Operation	Other expression
Przykład	tcp	dst	10.10.1.1	80	and	tcp dst 10.10.2.2 3128

Protocol (Protokół):
Wartości: ether, fddi, ip, arp, rarp, decnet, lat, sca, moprc, mopdl, tcp and udp.
Jeśli nic nie zostanie zdefiniowane, użyte zostaną wszystkie występujące.

Direction (Kierunek):
Wartości: src, dst, src and dst, src or dst
Jeśli nie zdefiniowane jest źródło lub przeznaczenie, zostanie zastosowane "src or dst".
Np.: "host 10.2.2.2" jest tożsame z "src or dst host 10.2.2.2".

Host(s) (Maszyna główna):
Wartości: net, port, host, portrange
Jeśli nie zdefiniowane, użyte zostanie domyślnie ”host”.
NP.” "src 10.1.1.1" jest tożsame z "src host 10.1.1.1".

Logical Operations (Operacje logiczne):
Wartości: not, and, or.
Negacja [Logiczne "nie"] ("not") ma najwyższą wartość. Następstwo [Logiczne "lub"] ("or") oraz powiązanie [logiczne "i"] ("and") są równe sobie i łączą się z lewej do prawej.
Np.:
"not tcp port 3128 and tcp port 23" jest tożsame z "(not tcp port 3128) and tcp port 23".
"not tcp port 3128 and tcp port 23" jest różne od "not (tcp port 3128 and tcp port 23)".

Filtry wyświetlania służą do wyszukiwania danych wewnątrz zrzuconych z pomocą filtrów przechwytywania pakietów.
Możliwości tych filtrów są znacznie rozszerzone niż wcześniejszych. Definiując je nie musimy uruchamiać ponownie przechwytywania.

Składnia	Protocol	String 1	String 2	Comparison Operator	Value	Logical Operations	Other expression
Przykład	ftp	passive	ip	==	10.2.3.4	xor	icmp.type

Protocol (Protokół):
Dostępna jest duża liczba protokołów zlokalizowana jest pomiędzy warstwami 2 i 7 modelu OSI. Widoczne są one jeśli wybierzemy przycisk "Expression..." w oknie głównym.
Przykładami są: IP,TCP,DNS,SSH

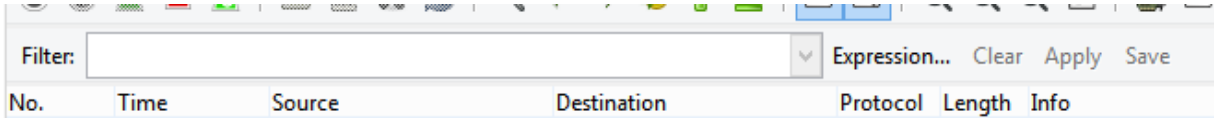
String1, String2 (Ustawienia opcjonalne):
Kategorie podprotokołów wewnątrz protokołu.
W celu ich znalezienia należy rozwinąć właściwości protokołu klikając w "+".

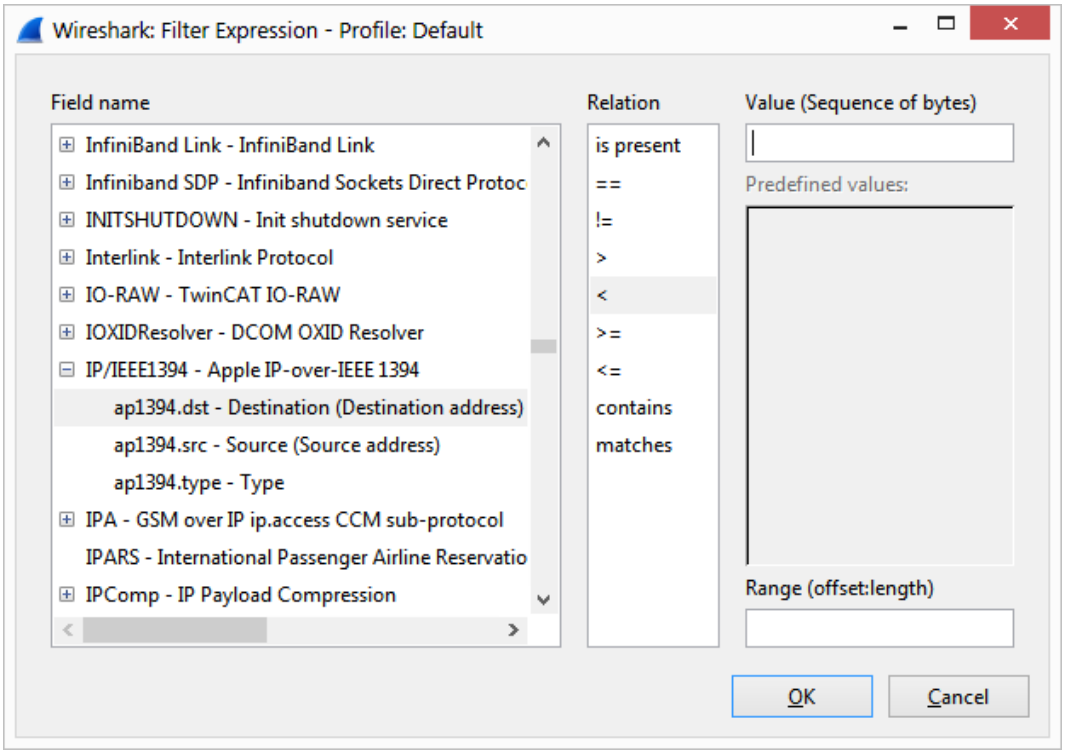
Comparison operators (Porównanie operatorów): :
Dostępne jest porównanie sześciu:

Format angielski:	Format w stylu C:	Znaczenie:
eq	==	Jest równe (equal)
ne	!=	Nie jest równe (Not Equal)
gt	>	Większe od (Greater than)
lt	<	Mniejsze od (Less than)
ge	>=	Większe lib równe (Greater or equal)
le	<=	Mniejsze lub równe (Less or equal)

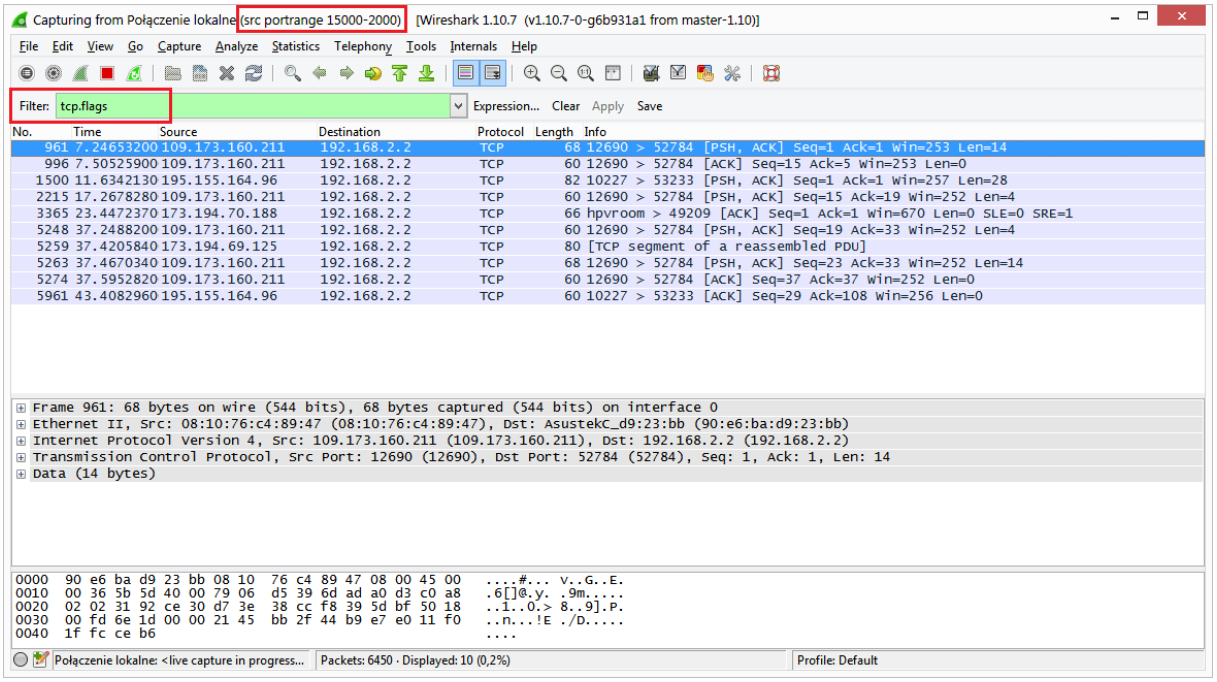
Logical expressions (Wyrażenia logiczne):

Format angielski:	Format w stylu C:	Znaczenie:
and	&&	Logical AND (Logiczne "I")
or	||	Logical OR (Logiczne "LUB")
xor	^^	Logical XOR (Logiczne "XOR")
not	!	Logical NOT (Logiczne zaprzeczenie)





Przykład wyników filtra przechwytywania „src portrange 15000-25000”, który wyłapuje pakiety z portów UDP lub TCP o zakresie 15000-25000 oraz zastosowanie filtra wyświetlania „tcp.flags”, który wyświetla tylko pakiety zawierające flage TCP.



Składnia filtrów przechwytywania jest taka sama jak w przypadku innych programów używających biblioteki Lipcap (Linux) lub Winpcap(Windows), jak znany TCPdump.

2. Ettercap

Drugim przydatnym narzędziem jest ettercap. Program ten rozwinął się z aktywnego sniffera, który potrafił przekierować ruch do interfejsu analizującego także w środowisku wykorzystującym przełączniki sieciowe. Opcja ta oczywiście nadal jest dostępna, stanowiąc część możliwości sniffera w trybie zunifikowanym (united sniffing), ale w tej chwili jedynie uzupełnia arsenał środków. Mocną stroną oprogramowania ettercap jest bridged sniffing, gdzie oprogramowanie zostaje uruchomione na hoście, który ma dwa interfejsy sieciowe, tworząc most. W ten sposób można analizować cały ruch wychodzący z konkretnej maszyny lub segmentu sieci, bez konieczności korzystania z tapa lub rekonfiguracji przełączników.

Ettercap to w zasadzie zestaw do ataków typu MiM (Man in the Middle) na sieci LAN. Dla tych którzy nie lubią interfejsu linii poleceń, jest wyposażony w prosty interfejs graficzny.

Ettercap jest w stanie przeprowadzić ataki przeciwko protokołowi ARP przez pozycjonowanie siebie jako „Człowiek po środku", i kiedy to zrobi jest zdolny do:

- zainfekować, podmienić, usunąć dane w połączeniu
- odkryć hasła dla protokołów takich jak FTP,http,POP,SSH1, itp.
- Dostarczyć fałszywe certyfikaty SSL w sekcjach HTTPS ofiar
- Itd..

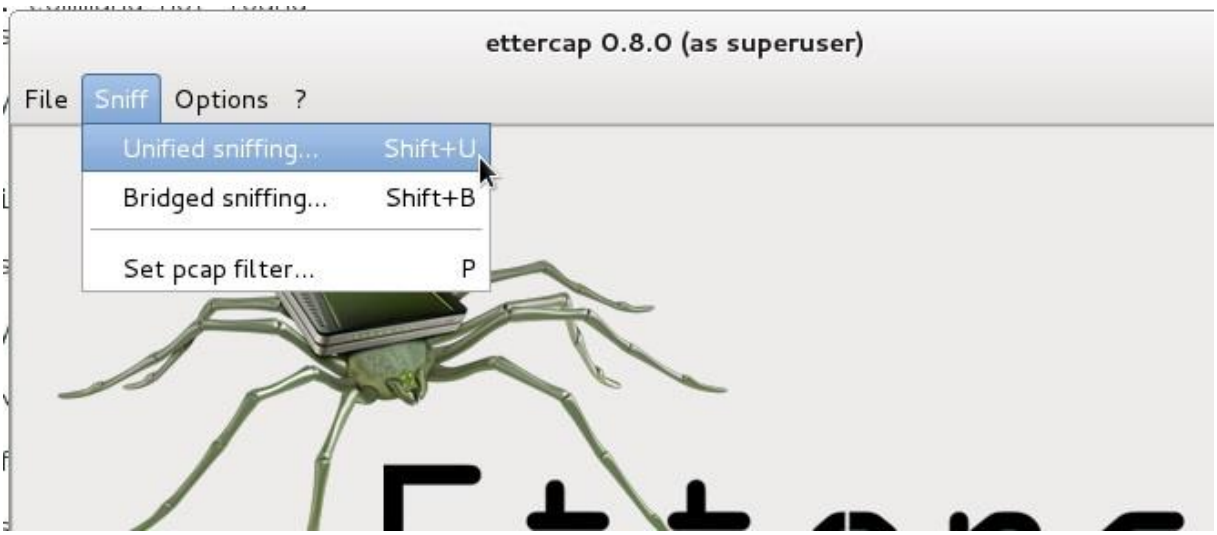
Przykład przeprowadzenia ataku typu ARP Spoofing, aby umieść aplikacje ettercap jako „Man in the middle".

Uruchamiamy program ettercap z interfejsem graficznym GTK

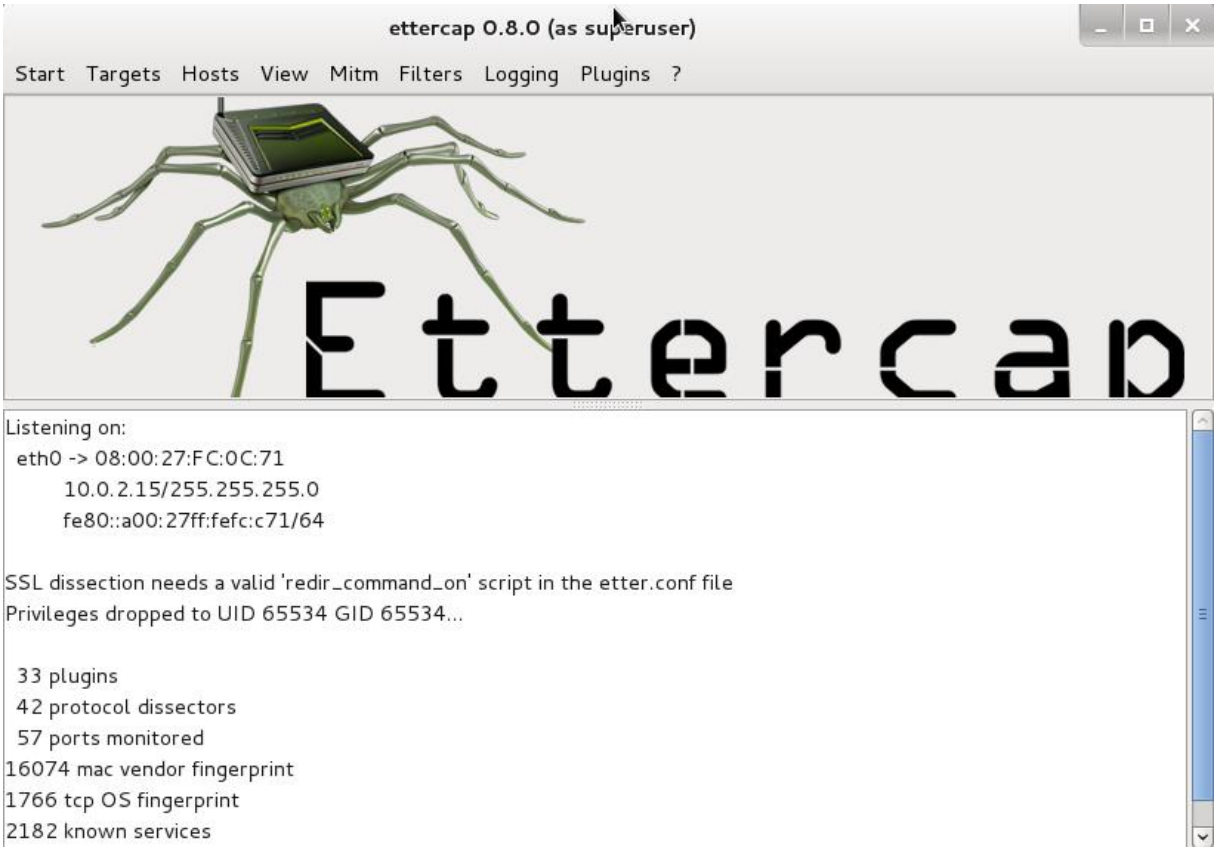
```
Setting up ethtool (1:3.4.2-1) ...
root@debian:/home/skwarek/Downloads/ettercap-0.8.0/build# ettercap -G

ettercap 0.8.0 copyright 2001-2013 Ettercap Development Team
```

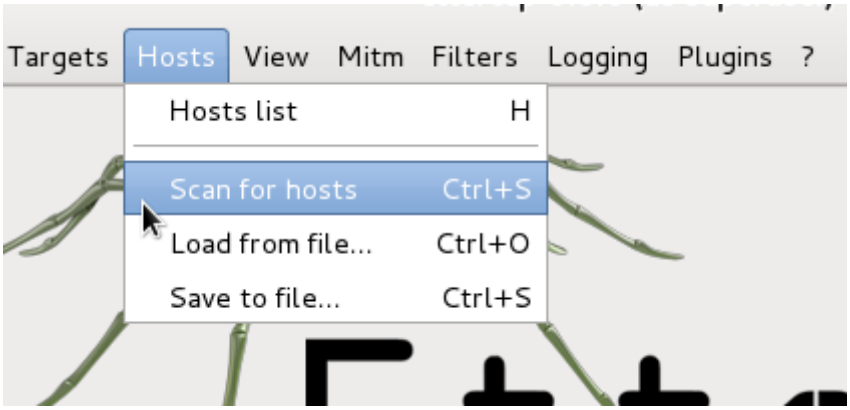
Następnie Skiff > Unified sniffing.. i wybieramy interfejs sieciowy

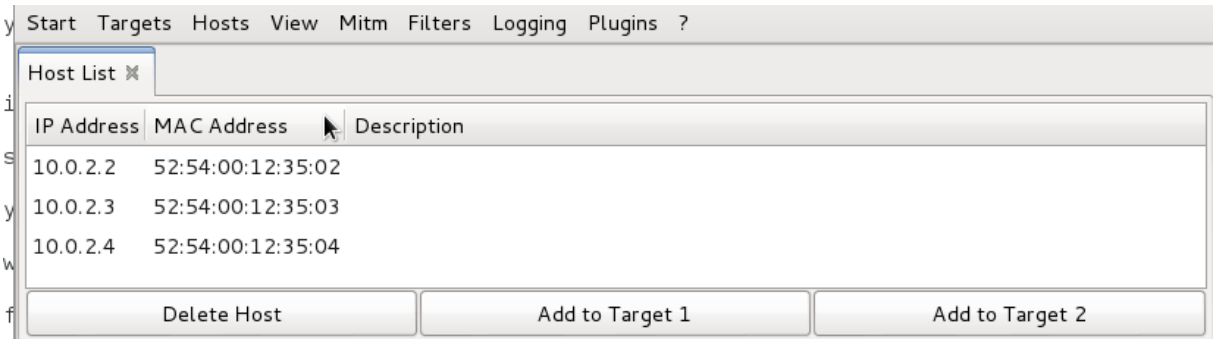


Potwierdzenie nasłuchiwania na interfejsie eth0

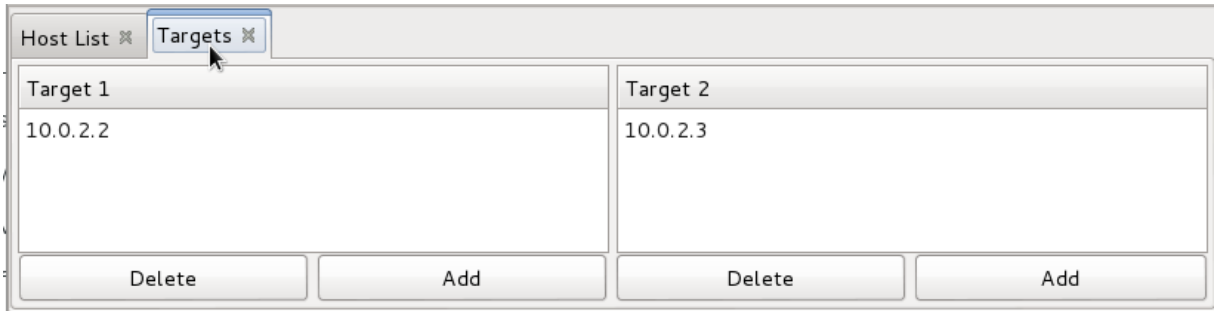


Wyszukiwanie hostów

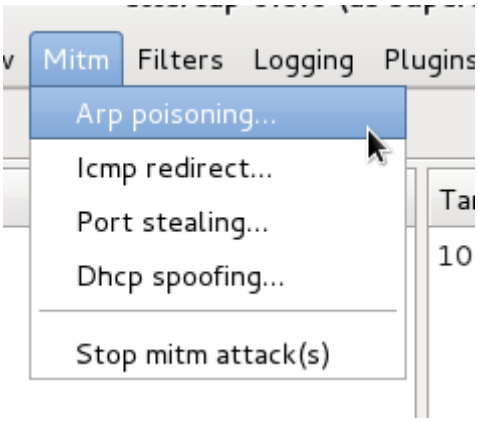


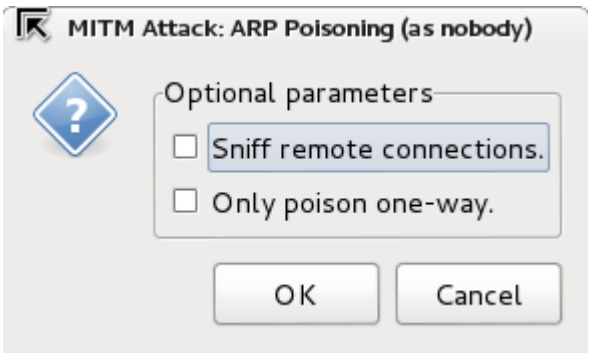


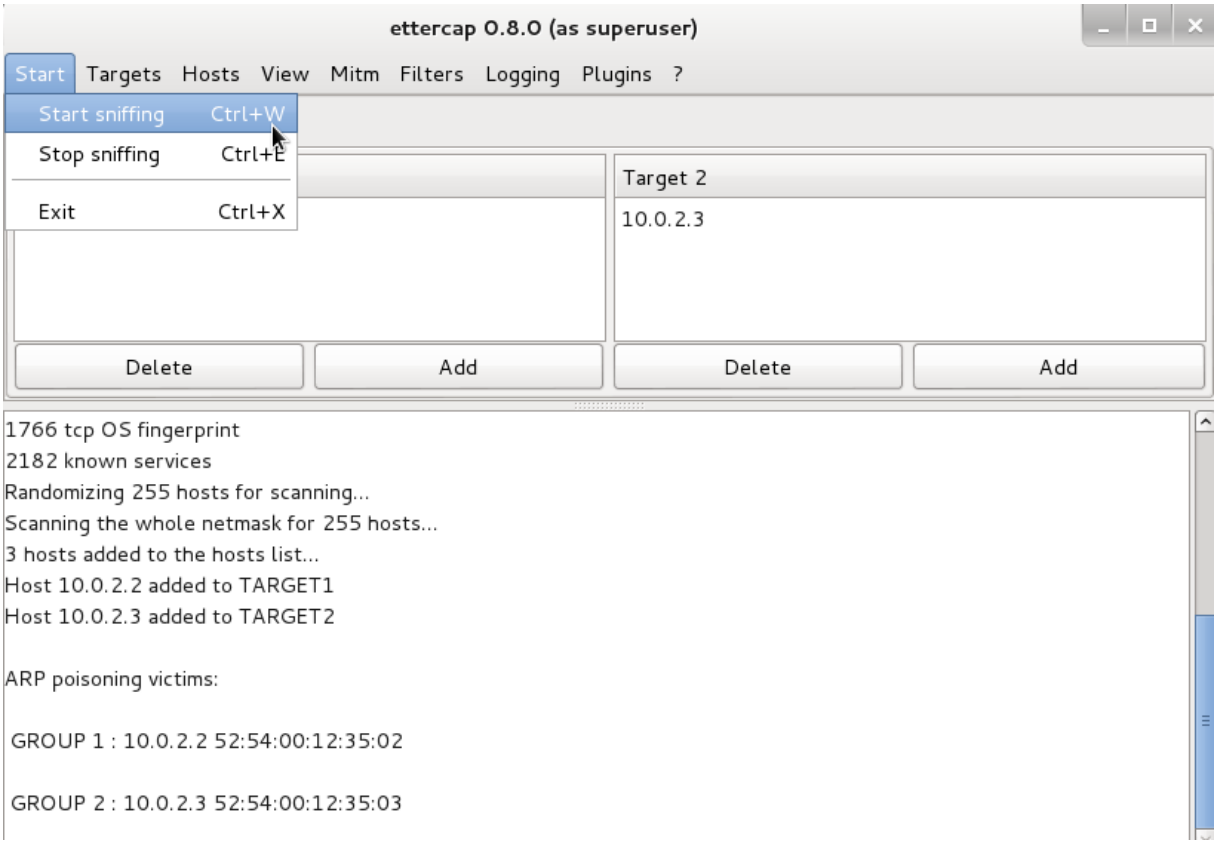
Wybór celu



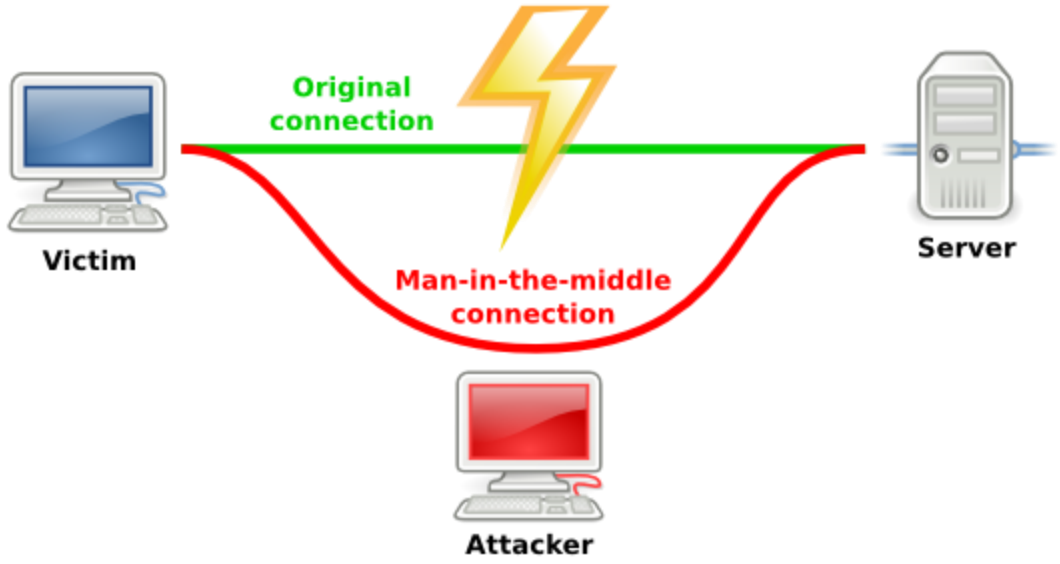
I w końcu rozpoczęcie ARP poisoning







Tym sposobem Ettercap stał się pośrednikiem połączenia pomiędzy wybranymi hostami. Tym samym może podmieniać lub usuwać informacje przesyłanymi miedzy nimi.



3. TCPdump

Tcpdump jest jednym z popularniejszych analizatorów sieciowych (potocznie nazywanych snifferami) pracujących na platformie uniksowej. Głównym zadaniem tcpdumpa jest nasłuchiwanie ruchu panującego w sieci. Można wykorzystać ten program w dwóch głównych celach takich jak: analiza pakietów, które przepływają przez sieć oraz w celu przechwytywania informacji wysyłanych przez innych użytkowników

Podstawowym parametrem z jakim możemy wywołać program, jest “-i” (od ang. interface) a następnie po nim podajemy nazwę interfejsu karty na którym ma nasłuchiwać, przykładowo może to być eth0 (pierwsza karta sieciowa). Nasz program z takim parametrem będzie nasłuchiwać wszystkie pakiety, które przechodzą przez ten interfejs. Kompletne wywołanie będzie wyglądać następująco:

```
root@debian:/home/skwarek# tcpdump -i eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
13:46:16.128798 IP cls9.any.onet.pl.https > debian.local.55131: Flags [P.], seq 18244404:18244431, ack 163486680
13:46:16.129286 IP debian.local.55131 > cls9.any.onet.pl.https: Flags [.], ack 27, win 28400, length 0
13:46:16.130055 IP cls9.any.onet.pl.https > debian.local.55131: Flags [F.], seq 27, ack 1, win 65535, length 0
13:46:16.158311 IP debian.local.55131 > cls9.any.onet.pl.https: Flags [F.], seq 1, ack 28, win 28400, length 0
```

Należy wspomnieć także o parametrze “-n”, którego zastosowanie jest bardzo pomocne, ponieważ dzięki niemu nie tracimy czasu na konwertowanie adresów IP na nazwy hostów. Kolejnym ważnym parametrem, z którym wywołujemy program to “-v”. Pozwala on na dokładniejszą analizę pakietów

```
ek# tcpdump -i eth0 -v
```

Zajmijmy się teraz wnętrzem pakietu, które można zobaczyć w postacie heksadecymalnej (szesnastkowej) używając parametru "-x" przykładowo:

```
root@debian:/home/skwarek# tcpdump -i eth0 -x
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
13:50:02.356531 IP debian.local.32850 > google-public-dns-a.google.com.domain: 19968+ A? google.pl. (27)
        0x0000:  4500 0037 3878 4000 4011 e61f 0a00 020f
        0x0010:  0808 0808 8052 0035 0023 1c53 4e00 0100
        0x0020:  0001 0000 0000 0000 0667 6f6f 676c 6502
        0x0030:  706c 0000 0100 01
```

Tcpdump może oprócz nasłuchiwania całego interfejsu sieciowego, przejść na tryb nasłuchiwania poszczególnych portów, protokołów albo ruchu pomiędzy danymi komputerami w sieci itd. Zacznijmy od pierwszej metody. Aby ustawić tcpdump na nasłuchiwanie konkretnego portu należy go uruchomić w następujący sposób:

```
# tcpdump -i eth0 port 80
```

Nasłuchiwanie konkretnego protokołu sieciowego:

```
# tcpdump -i eth0 icmp
```

Poniższy przykład pokazuje jak możemy podsłuchiwać dany port na zdefiniowanym komputerze w sieci.

```
# tcpdump -i eth0 src 10.0.2.15 and port 80
```

Oczywiście wyjście na konsolę nie zawsze jest wygodne - możliwe jest również zapisywanie zbieranych pakietów do pliku. Struktura pliku to właśnie legendarny format tcpdump, z którym zgodne są (tzn. potrafią go odczytać a często również zapisać dane w tym formacie) wszystkie liczące się sniffery (np. bardzo popularny Wireshark). Aby zatem zmusić tcpdump do logowania informacji do pliku, dodajemy parametr *„-w nazwa_pliku",* gdzie nazwa_pliku to jak nietrudno zgadnąć, miejsce gdzie program zapisze dane.