Wojciech Myszka

Laboratorium 2: Analiza ruchu w sieci albo jej podglądanie wer. 30 z drobnymi modyfikacjami!

2021-03-07 20:57:04 +0100

Spis treści

1. Wprowadzenie 2
1.1. Przełączniki sieciowe 2
1.2. Sniffery 3
2. Proste zadania do wykonania 4
3. Ogólne zasady uruchamiania programu tcpdump 9
3.1. Filtry 9
3.2. Interfejs 11
4. Zapis analizowanych informacji do pliku 11
5. Wireshark 12
6. Lektury dodatkowe 12
7. Instrukcja w postaci jednego pliku... 12

1. Wprowadzenie

W sytuacji normalnej dane podczas wysyłania do sieci przechodzą automatycznie przez wszystkie warstwy stosu TCP/IP. Podobnie jest z odbieranymi danymi. W modelu tym zakłada się, że karta sieciowa przyjmuje jedynie dane skierowane do niej (to znaczy wysyłane po określony adres IP, albo określony adres fizyczny). Obejmuje to oczywiście sytuację adresów rozgłoszeniowych (czyli złożonych z samych jedynek[1]).

Każda warstwa odbiera informacje, usuwa nagłówki i dodatkowe informacje i przekazuje pakiet wyżej. Na końcu dane są rozkodowywane (i ewentualnie prezentowane użytkownikowi) przez aplikację.

Można jednak kartę sieciową ustawić w taki tryb, alby odbierała wszystkie informacje pojawiające się w medium fizycznym. Tryb taki nazywa się *promiscuous mode* albo tryb nasłuchiwania (choć powinno pewno być „tryb podsłuchiwania").

1.1. Przełączniki sieciowe

W czasach prehistorycznych (sieć zbudowana w technologii kabla koncentrycznego) karta pracująca w tym trybie rzeczywiście mogła zbierać wszystkie informacje pojawiające się w medium.

Gdy nadeszły czasy skrętki — sytuacja powoli zaczęła się zmieniać. Najpierw używano koncentratorów (ang. *hub*), które były funkcjonowały w najniższej warstwie (fizycznej) duplikując sygnały sygnał pojawiający się na jednym porcie na pozostałych.

Hub to taka skrzyneczka zawierająca kilka-kilkanaście gniazdek sieciowych, do których podłącza się komputery.

[1] W przypadku adresów IP będą to same jedynki w polu przeznaczonym na adres węzła.

Wraz ze wzrastającym ruchem pojawiła się potrzeba separowania go. Huby zastąpione zostały przez przełączniki[2] (działając w drugiej warstwie sieciowej) i zachowujące się trochę jak centrala telefoniczna: łącząca nadawcę z odbiorcą informacji na czas trwania transmisji. Z zewnątrz przełącznik nie różni się od koncentratora, ale ma wbudowaną inteligencję i wie, do którego portu podłączona jest karta o określonym adresie fizycznym[3]. Dzięki temu może przekazywać do poszczególnych portów ruch sieciowy kierowany tylko do nich.

Tak więc karta sieciowa podłączona do koncentratora ma bardzo ograniczone możliwości nasłuchu ruchu sieciowego. Istnieją natomiast przełączniki, pozwalające duplikować ruch ze wszystkich (bądź wybranych) portów na wskazanym porcie, do którego podłączony jest komputer z oprogramowaniem pozwalającym na analizę ruchu sieciowego.

1.2. Sniffery

Sniffer to specjalne urządzenie lub specjalny program komputerowy, którego zadaniem jest przechwytywanie i analiza danych przepływających w sieci. Do najpopularniejszych programów tego typu należą:
— tcpdump,

[2] Po angielsku switche.

[3] Oczywiście do jednego portu przełącznika podłączonych może być (z wykorzystaniem koncentratora lub przełącznika) kilka komputerów. Wówczas na wyjściu pojawią się pakiety kierowane do któregokolwiek z tych komputerów.

— sniffit,
— ettercap,
— dsniff,
— wireshark (dawniej ethereal)
oraz
— snort.

Trudno powiedzieć, który jest najlepszy. **tcpdump** jest typowym, uniksowym narzędziem uruchamianym z linii polecenia, z parametrami przekazywanymi bezpośrednio z tej linii. **wireshark** to program graficzny, o rozbudowanym interfejsie i możliwościach zbliżonych do programu tcpdump.

— Elementarne instrukcje dostępne są na różnych „hakerskich" portalach sieciowych, na przykład tu: https://haker.edu.pl/2015/10/14/wireshark-sniffer-i-tcpdump-podsluch/.
— Dokumentacja do programu tcpdump z oficjalnych stron: https://www.tcpdump.org/#documentation.
— Dokumentacja programu wireshark z oficjalnej strony: https://www.wireshark.org/#learnWS[4].

Oba programy są zainstalowane w laboratorium. Studenci mają prawo uruchamiania ich.

2. Proste zadania do wykonania

Uwaga: Zadania powinny być wykonywane w warunkach „laboratoryjnych", to znaczy najlepiej zamknąć przeglądarkę, która potrafi żyć własnym życiem i nie robić równocześnie wielu rzeczy. Nic się nie zepsuje, ale wyniki będą znacznie trudniejsze do analizy.

1. Pakiet ping do wybranego węzła.
   Potrzebne będą dwa terminale[5]. W jednym z nich piszemy:

[4] Można też w sieci znaleźć kopie pdf książki Network Analysis using Wireshark Cookbook.

[5] Terminal otwieramy naciskając równocześnie trzy klawisze: Ctrl-Alt-T.

`sudo tcpdump icmp`
a w drugim;
`ping -c 1 156.17.8.1`
i obserwujemy wyniki. Pracę programu tcpdump kończymy naciskając równocześnie klawisze CTRL i C.
Doświadczenie można powtórzyć, prosząc tcpdump o więcej informacji:
`sudo tcpdump -v icmp`
Wyjaśnienia:
— przełączenie karty w tryb podsłuchiwania wymaga uprawnień administratora, stąd sudo;
— parametr icmp po nazwie programu to prośba o monitorowanie ruch protokołu ICMP;
— z wyników podsłuchiwania ważne są tylko dwie linijki;
— parametr -c polecenia ping ogranicza liczbę pakietów do jednego.

2. Jak działa program mtr (trzeba poczytać co to jest TTL czyli *Time To Live*):
`sudo tcpdump -v icmp`
a w drugim
`mtr -c 2 jsos.pwr.edu.pl`
Objaśnienia: parametr `-c 2` (tylko dwa obiegi) jest po to aby nie zaśmiecać wyników tcpdump zbyt wieloma informacjami.
Objaśnić otrzymane wyniki.
3. Połączenie UDP.
Protokół UDP jest prostszy niż TCP. Najlepiej widać to w przypadku zapytań DNS. W pierwszym oknie uruchamiamy tcpdump:
`tcpdump udp`
w drugim wpisujemy:
`host 156.17.8.1`
Trudno powiedzieć ile informacji pojawi się w oknie. Komputery prowadzą bardzo bujne życie (choć jest ono — zazwyczaj — ukryte przed naszym okiem). W przypadku kłopotów z analizą otrzymanych danych — patrz rozdział 4.

Interesuje nas linia zawierające `PTR? 1.8.17.156.in-addr.arpa.` To jest właśnie zapytanie o nazwę symboliczną węzła o podanym numerze. Trzeba zaobserwować numerek występujący przed PTR i musimy znaleźć inne linie zawierające ten numer. Oprócz znacznika czasu w linii są informacje o węźle wysyłającym zapytania i o węźle do którego kierowane jest zapytanie rozdzielone znakiem `>`. Przeanalizować zapytanie i uzyskane odpowiedzi.

4. Połączenie TCP.
   Tym razem zadanie jest trudniejsze (to znaczy generujące znacznie więcej informacji). W pierwszym oknie piszemy
   `tcpdump port 80`
   a w drugim
   `wget http://temisto.immt.pwr.wroc.pl/`
   Przed analizą otrzymanych wyników sugeruję lekturę informacji o nawiązywaniu i kończeniu połączeń TCP. (Tu trzeba uważać na pierwsze i ostatnie trzy pakiety.) Warto też coś poczytać o protokole HTTP.
   Później można przeanalizować wyniki śledzenia pakietów w przypadku połączenia szyfrowanego, w pierwszym oknie:
   `tcpdump port 443`
   w drugim oknie:
   `wget https://temisto.immt.pwr.wroc.pl/`
   Warto też zobaczyć co się stanie gdy zażądamy połączenia nieszyfrowanego z serwerem kmim.wm.pwr.edu.pl:
   `tcpdump port 80`
   a w drugim (wybieramy inny serwer, bo kmim realizuje **tylko** połączenia szyfrowane):
   `wget http://wm.pwr.edu.pl/`
   (teraz http, a nie https).
   Powyższe ćwiczenie należy powtórzyć uruchamiając program tcpdump w trybie *verbose*:
   `tcpdump -v port 80`
   albo nawet:
   `tcpdump -vv port 80`

(numer portu trzeba zmieniać adekwatnie do sytuacji, to znaczy na 443 w przypadku https).
Objaśnienia:
— `wget http://wm.pwr.edu.pl/` uruchamia program przeznaczony do ściągania i zapisywania na dysku stron WWW. Działa jak przeglądarka, ale strony nie wyświetla, nie ściąga też obrazków ani żadnych dodatkowych plików.

Opisać i wyjaśnić różnice w przypadku połączenia szyfrowanego i nie.

5. Analiza danych przekazywanych w pakietach.
Powyższe zapytania dawały dosyć ogólne informacje. Tylko najważniejsze — z punktu widzenie komunikacji — informacje były wyświetlane. Można poprosić o wyświetlenie zawartości pakietów (w miejsce `xxx.xxx.xxx.xxx` wpisujemy adres IP komputera, przy którym siedzimy):

```
tcpdump -nvXSs 0 src xxx.xxx.xxx.xxx and
                      dst port 80 or src port 80
```

lub lepiej (patrz rozdział 4):

```
tcpdump -nvXSs 0 src xxx.xxx.xxx.xxx and
                     dst port 80 or src port 80 -w nazwa.pliku
```

(polecenia wpisujemy w jednej linii).
W drugim oknie, podobnie jak w zadaniu 4 ściągamy ulubioną stronę www korzystając z protokołu http, a później https. Pamiętamy o portach: 80 dla http i 443 dla https!
Objaśnienia parametrów tcpdump:
— `-n` — nie konwertuj adresów IP, numerów portów do postaci symbolicznej,
— `-v` — wypisuj nieco więcej informacji,
— `-X` — oprócz nagłówków drukuj zawartość pakietów (szesnastkowo i w ASCII),
— `-S` – drukuj numery pakietów w formie bezwzględnej a nie względnej,
— `-s` — z każdego pakietu drukuj tylko określoną liczbę bajtów; `-s 1000` oznacza drukuj jedynie pierwszych 1000 bajtów

z każdego pakietu; `-s 0` — nie ograniczaj liczby drukowanych bajtów: drukuje wszystko.

Zamiast wpisywać parametry osobno: `-n -v -X -S -s 0` można wszystko połączyć: `-nvXSs 0`.

Podobnie jak poprzednio porównać wyniki z połączenia szyfrowanego i nie.

6. Podglądanie hasła.

   To jest raczej trudne zadanie — chyba już nie da się znaleźć stron na których wpisane hasła przekazuje się otwartym tekstem. Ale pewno takie są. W każdym razie przygotowałem taką stronę. Znajduje się ona pod adresem http://temisto.immt.pwr.wroc.pl/~myszka/auth/ w wersji nieszyfrowanej i https://temisto.immt.pwr.wroc.pl/~myszka/auth/ w wersji szyfrowanej.

   Wykorzystując umiejętności zdobyte w zadaniach 4 i 5 odnaleźć w danych nazwę użytkownika i hasło. Opisać procedurę.

   Pod (archiwalnym) adresem https://web.archive.org/web/20180517093344/https://wroot.org/posts/quick-and-dirty-tcpdump-credential-usernamepassword-sniffer/ znaleźć można bardzo prostą receptę na wyszukiwanie haseł w analizowanych danych.

```
tcpdump port http or port ftp or port smtp or port imap
        or port pop3 -l -A |
        egrep -i 'pass=|pwd=|log=|login=|user=
                  |username=|pw=|passw=|passwd=
                  |password=|pass:|user:|username:
                  |password:|login:|pass |user '
                 --color=auto --line-buffered -B20
```

wszystko trzeba wpisać w jednej linii albo dodać znaki backslash na końcu każdej:

```
tcpdump port http or port ftp or port smtp or port imap \
        or port pop3 -l -A | \
        egrep -i 'pass=|pwd=|log=|login=|user=\
                  |username=|pw=|passw=|passwd=\
```

```
|password=|pass:|user:|username:\
|password:|login:|pass |user '\
--color=auto --line-buffered -B20
```

Objaśnienie: program tcpdump podsłuchuje ruch w sieci (ograniczając się jedynie do takich przypadków, gdzy hasło może być przesyłane w postaci nieszyfrowanej: protokoły ftp, http, imap, pop3, smtp), zawartość pakietów konwertuje do ASCII (`-A`) i wszystko wysyła na standardowe wyjście, które przechwytuje program egrep (o operacji pipe `|` można trochę tu poczytać).
Program egrep wyszukuje korzystając z wyrażeń regularnych określonych ciągów znaków.
Sprawdzić czy to działa w tym przypadku.

3. Ogólne zasady uruchamiania programu tcpdump

3.1. Filtry

tcpdump jest dosyć prostym narzędziem. W linii polecenia definiuje się filtr, który ma dokładniej sprecyzować, które pakiety nas interesują. Elementami wyrażenia definiującego filtr mogą być różne warunki, można warunki łączyć za pomocą wyrażeń logicznych. Strona dokumentacji pcap-filter zawiera dokładniejsze informacje o zasadach tworzenia wyrażeń. Podstawowe informacje i przykłady zawiera strona dokumentacji tcpdump.

Strony manuala można czytać również lokalnie. W otwartym oknie terminala trzeba napisać:

```
man tcpdump
```

albo:

```
man pcap-filter
```

Filtr może definiować:

— adres[6] źródłowy (na przykład `src io.immt.pwr.wroc.pl`),
— adres docelowy (na przykład `dst 156.17.8.1`),
— numer portu (na przykład `port 80`),
— protokół:
  — ether,
  — fddi,
  — tr,
  — wlan,
  — ip,
  — ip6,
  — arp,
  — rarp,
  — decnet,
  — tcp
    i
  — udp.
— Rodzaj protokołu IP:
  — icmp,
  — icmp6,
  — igmp,
  — igrp,
  — pim,
  — ah,
  — esp,
  — vrrp,
  — udp,
    lub
  — tcp
— Można wskazać węzeł/węzły, z którymi interesuje nas komunikacja:
  `tcpdump host piwo017`
  natomiast zapytanie
  `tcpdump host piwo017 and \( io or piwo000 \)`

[6] Adresy można definiować numerycznie lub symbolicznie

będzie monitorowało wyłącznie komunikację węzłą piwo017 z węzłami io albo piwo000.

W zapytaniach filtra można odnosić się do zawartości nagłówków IP, ether,... Ale to dla hakerów.

3.2. Interfejs

W przypadku, gdy komputer ma kilka interfejsów sieciowych program trzeba uruchamiać z parametrem mówiącym, na którym z nich program ma pdsłuchiwać:

```
tcpdump -i eno1
```

(eno1 to podstawowy interfejs w laboratorium, w innym przypadku trzeba zastąpić odpowiednią nazwą; nazwy interfejsów ujawnia polecenie `ifconfig`.)

4. Zapis analizowanych informacji do pliku

Czasami gdy program produkuje strasznie dużo wyników, można zapisać je do pliku:

```
tcpdump -w nazwa.pliku
```

specyfikujemy, oczywiście, w linii poleceń jakie informacje nas interesują. Program nie wyprowadza nic na terminal, podając tylko liczbę zebranych pakietów. Do pliku zapisywane są pakiety w formie surowej. Zdekodować je można za pomocą programu tcpdump w trybie czytania z pliku:

```
tcpdump -r nazwa.pliku
```

Na przykład (zadanie numer 5)

```
tcpdump -nnvXSs 0 src xxx.xxx.xxx.xxx and
                  dst port 80 or src port 80  -w nazwa.pliku
```

(zamiast `xxx.xxx.xxx.xxx` wpisujemy adres IP własnego komputera).

5. Wireshark

Wszystkie zadania powinno się dać wykonać używając programu wireshark (również jest zainstalowany). Nie mam jednak siły opisywać jak wszystko to wyklikać w interfejsie graficznym.

6. Lektury dodatkowe

1. dsniff (Dosyć już stary zestaw narzędzi, który jednak ciągle może być przydatny.)
2. Atak „man in the middle”.
3. I jeszcze jeden wariant.
4. How to Conduct a Simple Man-in-the-Middle Attack.

7. Instrukcja w postaci jednego pliku...

...jest również dostępna.