Sieci komputerowe
Warsztaty 6

Mateusz Markiewicz

16 maja 2020

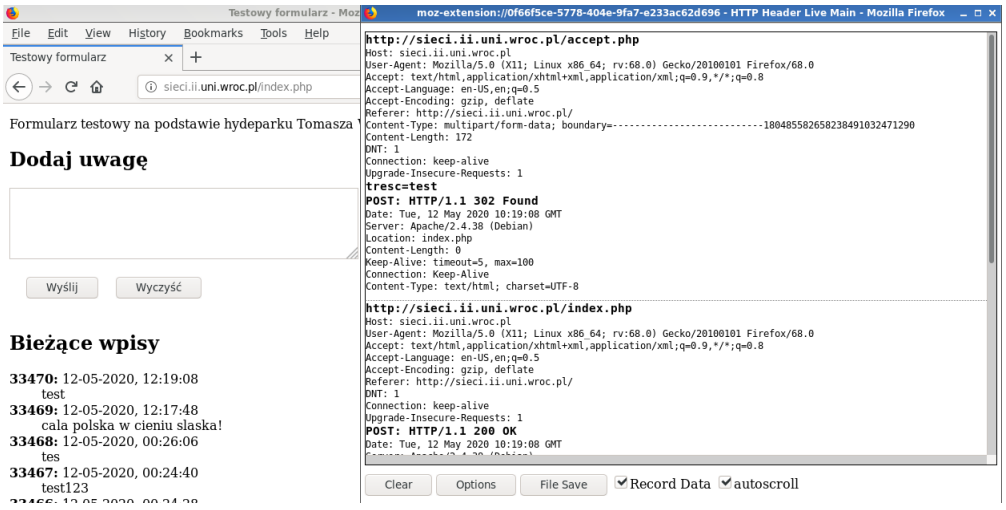
1 Pierwsze zadanie do zaprezentowania

Pierwszym etapem zadania było stworzenie maszyny wirtualnej z domyślną konfiguracją sieciową.
Następnie nazwałem interfejs tej maszyny, użyłem w tym celu następujących poleceń:

- V1#> ip link set enp0s3 name enp0
- V1#> dhclient -v enp0

Następnie do pliku */etc/hosts* za pomocą vima dodałem nowy wiersz:
156.17.4.30 sieci.ii.uni.wroc.pl

Następnie wszedłem na stronę http://sieci.ii.uni.wroc.pl/, uruchomiłem rozszerzenie *Live HTTP Header* oraz dodałem nowy wpis.

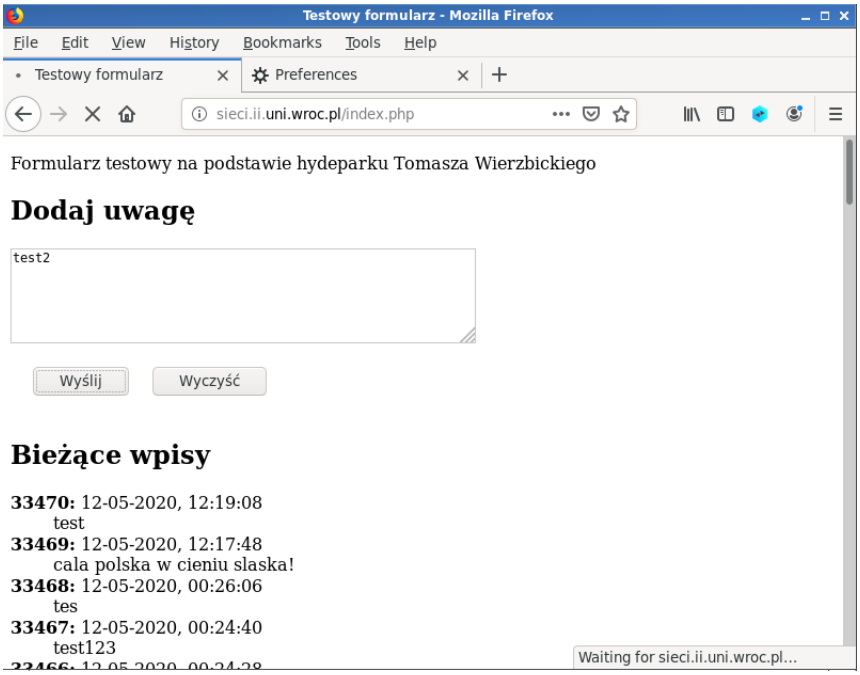


Dzięki rozszerzeniu możemy zaobserwować wysłane zapytanie HTTP (Post, z treścią, którą wysłaliśmy) oraz otrzymaną odpowiedź (200 OK).

Następnie uruchomiłem program nc w trybie serwera TCP nasłuchującego na porcie 8888 za pomocą polecenia:

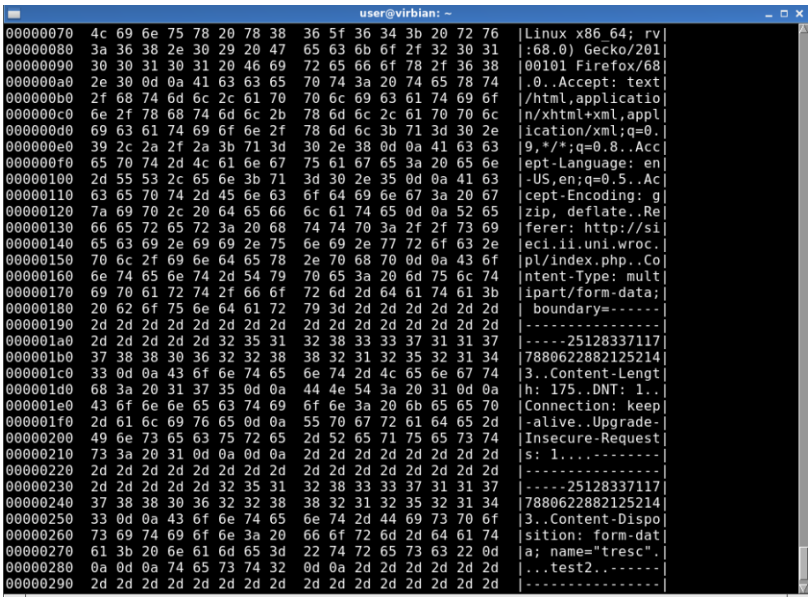
- V1$> nc -l -p 8888 | tee http_request

oraz w przeglądarce ustawiłem manualną konfigurację proxy pod adres localhost oraz port 8888 (czyli adres i port na którym działa nasz serwer TCP). Następnie spróbowałem dodać następny wpis na stronie http://sieci.ii.uni.wroc.pl/:



Jak widać przeglądarka wyświetla "Waiting for sieci.ii.uni.wroc.pl...". Dzieje się tak z powodu konfiguracji proxy, którą ustawiliśmy. Przeglądarka wysyła zapytanie HTTP pod adres localhist oraz port 8888. Zapytanie to dociera do naszego serwera TCP, ale nie dociera do serwera obsługującego formularz, dlatego wpis nie został dodany.

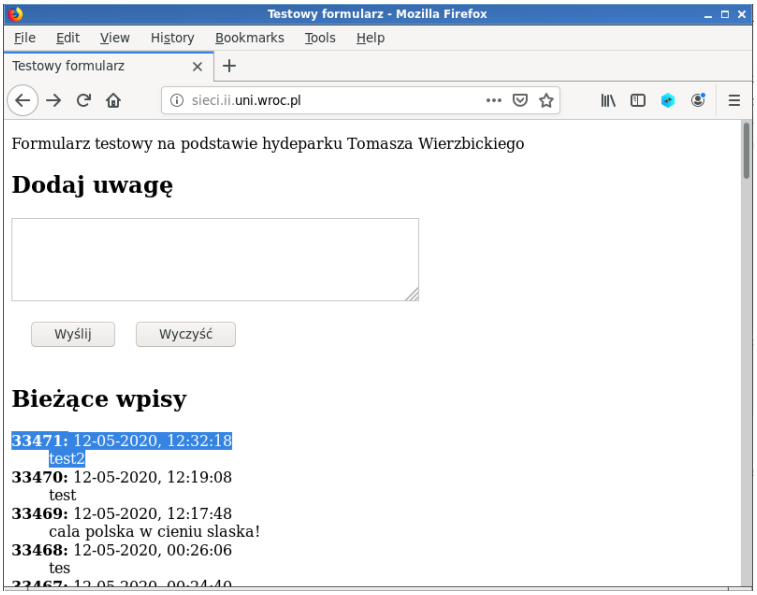
Następnie wyświetliłem zawartość pliku http_request:



Jak widzimy znajduje się w nim zapytanie HTTP. Możemy przesłać je do sieci.ii.uni.wroc.pl za pomocą polecenia:

- V1$> nc -q 3 sieci.ii.uni.wroc.pl 80 < http_request

po czym możemy zaobserwować, że wpis został dodany.



Następnie za pomocą vima edytowałem zawartość zapytania HTTP, po czym ponownie je wysłałem za pomocą tego samego polecenia co wcześniej. Ten wpis również został dodany.

2 Drugie zadanie do zaprezentowania

Na początku drugiego zadania za pomocą polecenia:

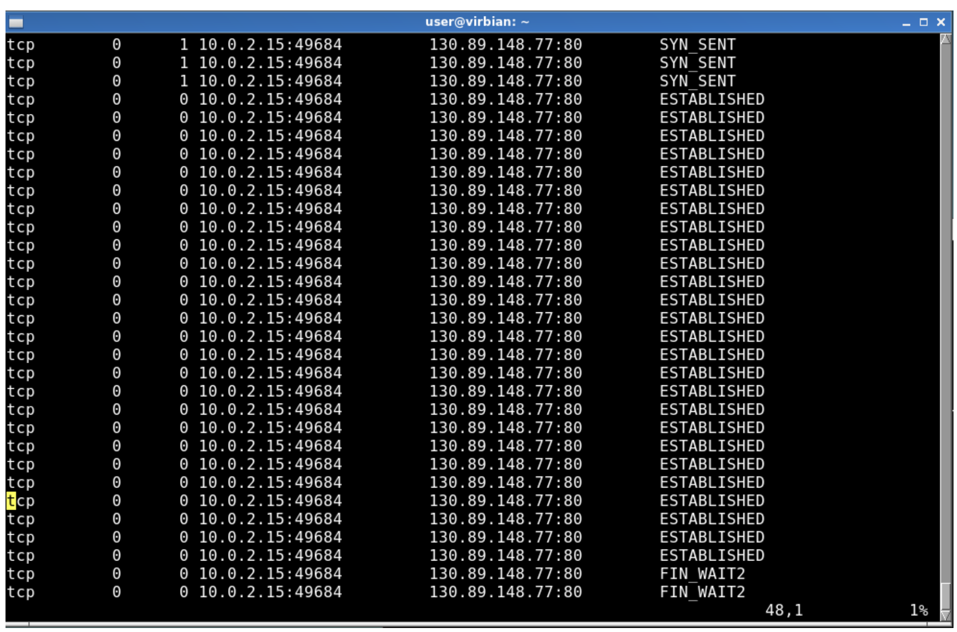
- V1$> dig www.debian.org

sprawdziłem adres IP domeny www.debian.org., który wynosi 130.89.148.77. Następnie w dwóch różnych konsolach uruchomiłem polecenia:

- V1C1$> (while true; do netstat -tan | grep 130.89.148.77 ; done) | tee tcp_log

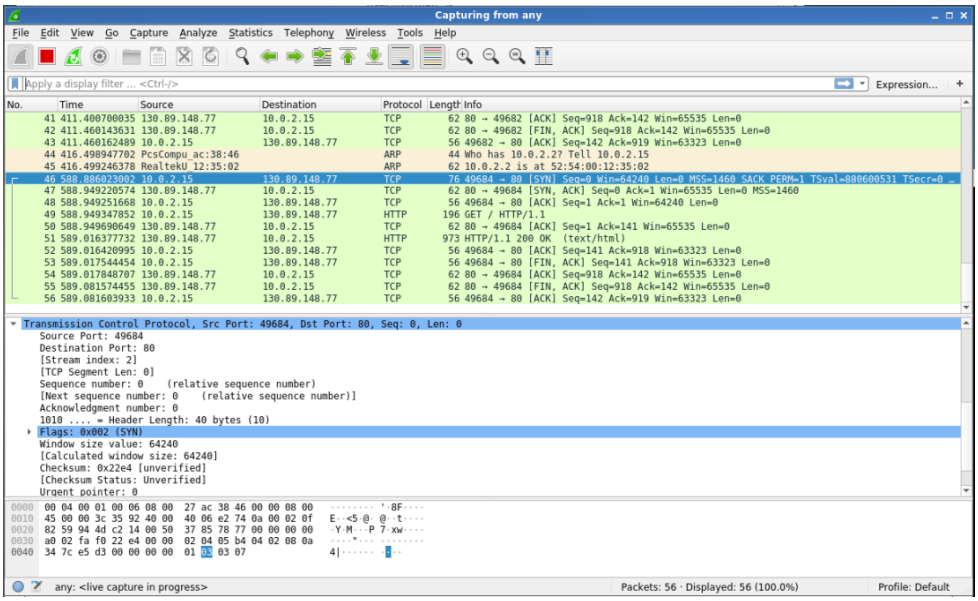
- V1C2$> wget http://130.89.148.77/

W rezultacie plik tcp_log miał następującą zawartość:



```
user@virbian: ~
tcp        0      1 10.0.2.15:49684         130.89.148.77:80        SYN_SENT
tcp        0      1 10.0.2.15:49684         130.89.148.77:80        SYN_SENT
tcp        0      1 10.0.2.15:49684         130.89.148.77:80        SYN_SENT
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        ESTABLISHED
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        FIN_WAIT2
tcp        0      0 10.0.2.15:49684         130.89.148.77:80        FIN_WAIT2
                                                                      48,1          1%
```

Za pomocą Wiresharka sprawdziłem jak wyglądały kolejne pakiety:



W celu pobrania strony tworzone jest gniazdo na podstawie adresu IP 10.0.2.15 oraz portu 49684, jest to port źródłowy połączenia. Portem docelowym jest 80. Przyjrzyjmy się każdemu wysłanemu / otrzymanemu segmentowi TCP:

- Pierwszy segment zostaje wysłany od nas do 130.89.148.77, ma flagę SYN, wysyłane są bajty od 0. Po stronie klienta połączenie TCP przechodzi w stan SYN SENT. Po tym jak server otrzyma ten segment połączenie TCP po stronie serwera przejdzie w stan SYN RECEIVED.
- Drugim segmentem jest odpowiedź otrzymana od 130.89.148.77, która ma flagi SYN oraz ACK. Wysyłane są bajty od 0, potwierdzane są bajty do 1. Połączenie po stronie klienta przechodzi w stan ESTABLISHED.
- Trzeci segment zostaje wysłany do 130.89.148.77, ma flagę ACK, przesyła bajty od 1, potwierdza bajty do 1. Po tym jak server otrzyma ten segment jego połączenie również przejdzie w stan ESTABLISHED.
- Następnie wysłane zostaje zapytanie HTTP oraz otrzymujemy segment od 130.89.148.77, który ma flagę ACK. Przesyła bity od 1, oraz potwierdza bity do 141. Stan połączenia po obu stronach pozostaje bez zmian.
- Następnie otrzymujemy odpowiedź HTTP oraz wysyłamy segment do 130.89.148.77 z flagą ACK, wysyłamy bajty od 141, a potwierdzamy bajty do 918. Po tym jak server otrzyma ten segment jego połączenie przejdzie w stan CLOSE WAIT.

- Kolejny segment jest wysłany przez nas do 130.89.148.77 z flagami ACK oraz FIN. W tym momencie zaczynamy zamykać połączenie. Połączenie po stronie klienta przechodzi w stan FIN WAIT 1.

- Następne dwa segmenty przychodzą od 130.89.148.77, pierwszy z nich ma flagę ACK, drugi ma flagi FIN i ACK. W obu widzimy, że Seq = 918, oraz ACK = 142, czyli server wysłał wszystko do bajtu 918 oraz otrzymał wszystko do bajtu 142. Po otrzymaniu pierwszego segmentu połączenie po stronie klienta przechodzi w stan FIN WAIT 2, po otrzymaniu drugiego segmentu połączenie to przechodzi w stan TIME WAIT

- Ostatni segment jest wysłany do 130.89.148.77 z flagą ACK, potwierdza on, że otrzymaliśmy wszystko do bajtu 919. Po tym jak server otrzyma ten segment połączenie po jego stronie przejdzie w stan CLOSED. Po upływie timeouta połączenie po stronie klienta również przejdzie w stan CLOSED.

Wynika z tego, że strona klienta wykonuje aktywne otwarcie oraz aktywne zamknięcie. W pliku tcp_log mogliśmy zaobserwować wszystkie stany połączenia po stronie klienta z wyjątkiem FIN WAIT 1.