Laboratorium 7

Wireshark cz. 4

W sprawozdaniu zamieść zrzuty ekranu z istotnych etapów oraz odpowiedzi na pytania.

Taki symbol oznacza, że trzeba w sprawozdaniu dodać zrzut ekranu (najczęściej 1) z wyniku działania polecenie.

Taki symbol oznacza, że należy dodać opis (najczęściej 1 zdanie) z wyniku działania polecenia.

I. Editcap

Editcap to jedno z kluczowych narzędzi wchodzących w skład pakietu Wireshark. Jego podstawowym zadaniem jest edycja i przetwarzanie plików PCAP oraz PCAPng. Umożliwia m.in. dzielenie dużych plików przechwytywania na mniejsze części, zmianę znaczników czasu pakietów czy filtrowanie ich według numerów sekwencyjnych.

1. Uruchom maszynę wirtualną z Kali Linux i przejdź do terminala.

2. Przechwyć ruch sieciowy na wybranym interfejsie i zapisz go do pliku `input.pcap`.

Wygeneruj tyle ruchu, aby plik zawierał ponad 10 000 pakietów.

3. Sprawdź zawartość utworzonego pliku Wiresharkiem lub:

```
tshark -r input.pcap
```

4. Podziel plik input.pcap na mniejsze pliki po 5000 pakietów. Każdy plik będzie miał nazwę rozpoczynającą się od `segment_numer`:

```
editcap -c 5000 input.pcap segment_numer
```

5. Sprawdź, czy pliki zostały utworzone:

```
ls -l
```

6. Sprawdź zawartość dowolnego segmentu używając tshark lub Wiresharka.

7. Przefiltruj pierwsze 500 pakietów z `input.pcap` i zapisz do pliku `filtrowanie.pcap`:

```
editcap -r input.pcap filtrowanie.pcap 1-500
```

8. Sprawdź zawartość pliku `filtrowanie.pcap`.

9. Odczytaj numer ostatniej ramki w pliku `input.pcap`, a następnie dopisz do niej komentarz i zapisz do pliku `komentarze.pcap`:

```
editcap -a "numer_ramki:Ostatnia przechwycona ramka mój_nr_indeksu" input.pcap komentarze.pcap
```

10. Wyświetl ramki z komentarzami:

```
shark -r komentarze.pcap -Y frame.comment -V
```

11. Przytnij plik do 10 000 pakietów i zapisz do `filtrowanie_3.pcap`.

```
editcap -d input.pcap filtorwanie_3.pcap 10001-x
```

x – numer ostatniego pakietu w pliku

12. Sprawdź, czy operacja się powiodła.

13. Wyodrębnij pakiety nr 1, 5, 10-20, 30-40 i zapisz do pliku filtrowanie_4.pcap:

```
editcap -r input.pcap filtrowanie_4.pcap 1 5 10-20 30-40
```

14. Sprawdź, czy operacja się powiodła.

Jedną z najbardziej kluczowych operacji jest wyodrębnienie z pliku `.pcap` pakietów sieciowych, które zostały przesłane w konkretnym przedziale czasowym, np. od 10:00:00 do 11:00:00 dnia 30.01.2024. Tego typu operacja jest szczególnie przydatna podczas analiz incydentów bezpieczeństwa, gdy znany jest przypuszczalny czas zdarzenia.

15. Zapisz wybrany przedział czasowy do pliku `filtrowanie_5.pcap`. Skorzystaj z pomocy editcap w celu przygotowania komendy:

```
editcap --help
```

II. Lua

Lua to elastyczny i wydajny język skryptowy, który pozwala na tworzenie zaawansowanych funkcji do rozkładania skomplikowanego ruchu sieciowego na zrozumiałe informacje (tzw. dissektory), filtrów oraz narzędzi analitycznych bezpośrednio w Wiresharku, co jest nieocenione przy badaniu podejrzanych zachowań sieciowych i danych przesyłanych przez potencjalnie złośliwe oprogramowanie.

Specjaliści ds. bezpieczeństwa mogą tworzyć skrypty wykrywające specyficzne wzorce, sygnatury komunikacyjne lub anomalie w ruchu sieciowym, które często towarzyszą złośliwemu oprogramowaniu, takie jak nietypowe zapytania DNS, nieoczekiwane połączenia na rzadko używanych portach czy nietypowe wzorce przepływu danych.

1. Uruchom terminal w systemie Kali Linux.

2. Utwórz plik ze skryptem Lua poleceniem:

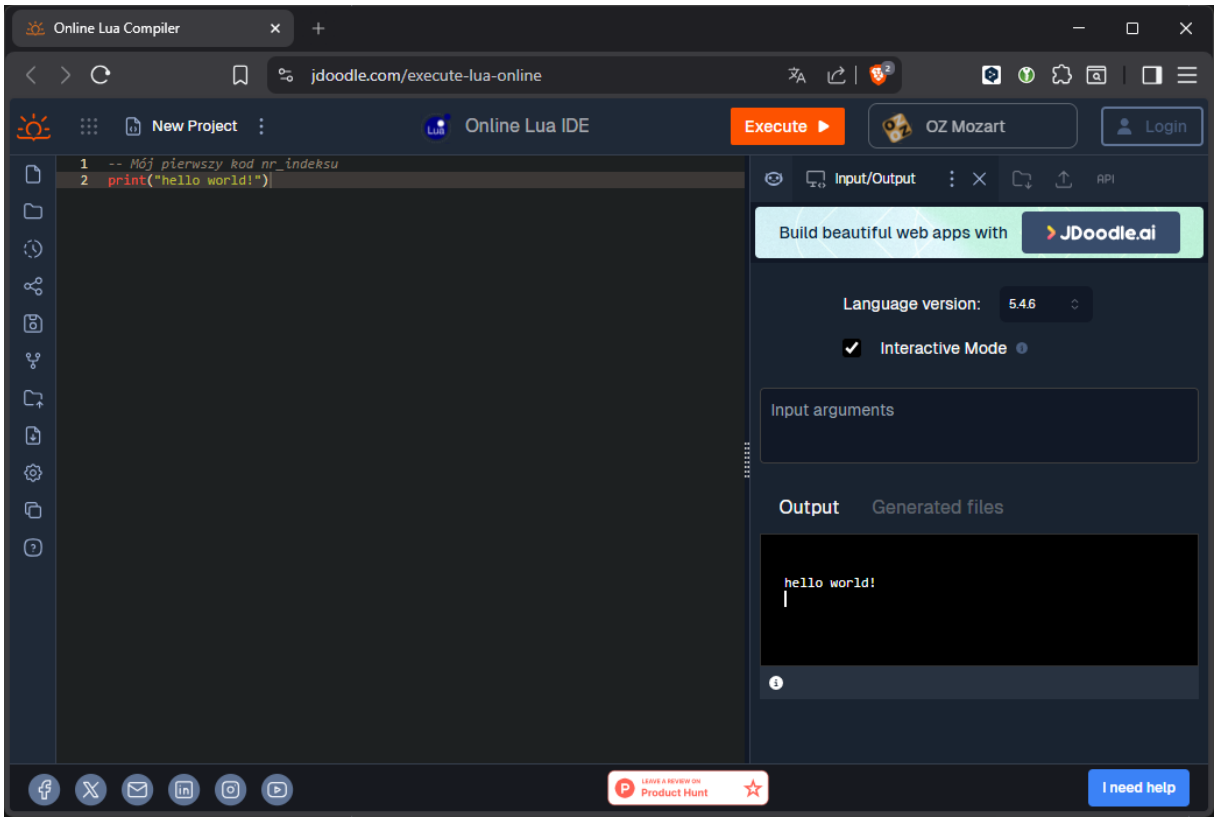
```
nano hello.lua
```

3. Wstaw do pliku następujący kod:

```
-- Mój pierwszy kod nr_indeksu
print("hello world!")
```

4. Przed pierwszym uruchomieniem warto sprawdzić, czy skrypt nie zawiera błędów składni. Przejdź do Lua Parser.

5. Wklej kod i wybierz przycisk `Execute`. Jeśli składnia jest poprawna – wynik pojawi się w polu `Output`. W przypadku błędu – `Output` wyświetli odpowiedni komunikat diagnostyczny.



6. Jeśli skrypt jest prawidłowy, zapisz plik i uruchom go poleceniem:

```
tshark -X lua_script:hello.lua
```

```
┌──(kali㉿kali)-[~]
└─$ tshark -X lua_script:hello.lua
hello world!
Capturing on 'eth0'
    1 0.000000000 192.168.217.133 → 146.59.45.208 NTP 90 NTP Version 4, client
    2 0.015927071 146.59.45.208 → 192.168.217.133 NTP 90 NTP Version 4, server
^C2 packets captured
```

Po wykonaniu skryptu tshark zacznie przechwytywanie więc przerwij je ctrl + c

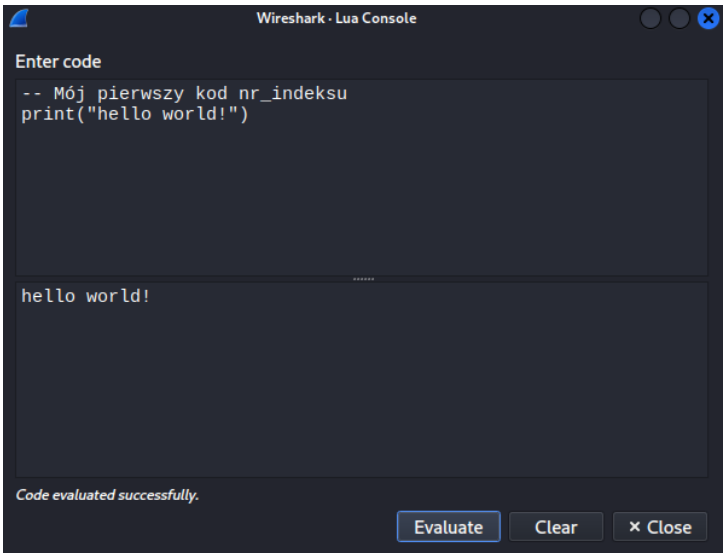
Drugim sposobem na uruchomienie skrypyu jest zrobienie to bezpośrednio w Wiresharku

7. Uruhcom Wireshark. Wpisz w terminalu

```
wireshark
```

8. Przejdź do Tools -> Lua Console

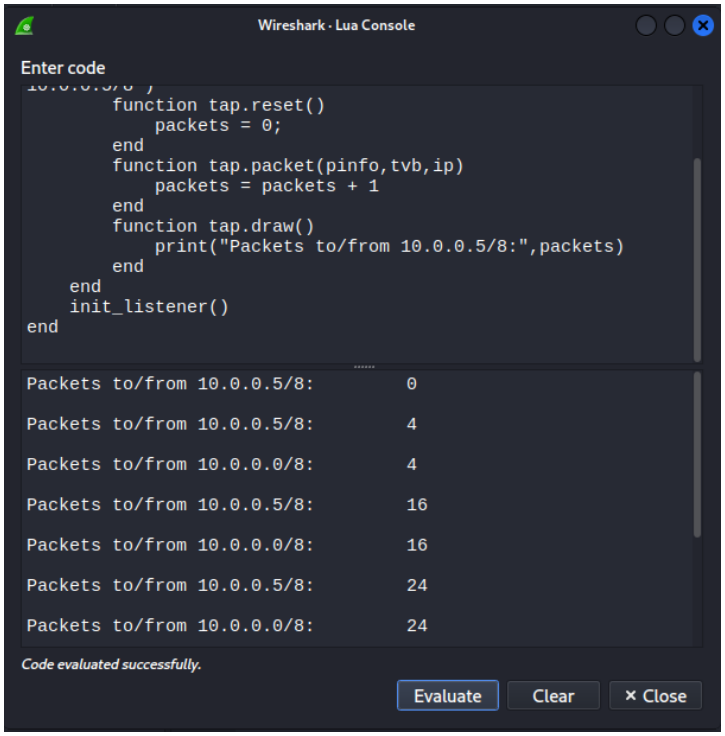
9. W oknie Lua Console wpisz kod w górnej części, a nastepnie nacisnij przycisk Evaluate



Utwórzmy coś bardziej użytecznego. Ponizszy skrypt tworzy listener, który zlicza pakiety spełniające określony warunek – w tym wypadku pakiety z/do adresu 10.0.0.5.

```
do
    packets = 0;
    local function init_listener()
        local tap = Listener.new("frame","ip.addr == 10.0.0.5/8")
        function tap.reset()
            packets = 0;
        end
        function tap.packet(pinfo,tvb,ip)
            packets = packets + 1
        end
        function tap.draw()
            print("Packets to/from 10.0.0.5/8:",packets)
        end
    end
    init_listener()
end
```

10. Uruchom powyższy skrypt na jeden z dwóch poznanych sposobów i przetestuj.



Lub

```
  261 15.154214963 192.168.217.133 → 142.250.109.94 TCP 54 54870 → 80 [ACK] Seq=435 Ack=1104 Win=64240 Len=0
  262 15.157124386 34.107.243.93 → 192.168.217.133 TCP 60 443 → 57226 [FIN, PSH, ACK] Seq=1188 Ack=2125 Win=64239 Len=0
  263 15.157228315 192.168.217.133 → 34.107.243.93 TCP 54 57226 → 443 [ACK] Seq=2125 Ack=1189 Win=63053 Len=0
  264 15.168495376 151.101.1.91 → 192.168.217.133 TLSv1.3 78 Application Data
  265 15.168515340 192.168.217.133 → 151.101.1.91 TCP 54 48070 → 443 [RST] Seq=937 Win=0 Len=0
^C265 packets captured
Packets to/from 10.0.0.5/8: 14

┌──(kali㉿kali)-[~]
└─$
```