

Paweł Kankowski 165764
Wojciech Borostowski 165375
Alexander Stiegler 165534

Wykorzystanie narzędzia docker do zobrazenia ataku typu man in the middle - Instrukcja

1. Wprowadzenie

Celem ćwiczenia jest zaprezentowanie ataku Man-in-the-Middle (MiTM) przy użyciu narzędzia Docker - otwartego oprogramowania do realizacji wirtualizacji na poziomie systemu operacyjnego, tworzącego kontenery ze skonfigurowanymi aplikacjami w środku.

Atak MiTM jest atakiem polegającym na wnikięciu atakującego lub złośliwych narzędzi pomiędzy komputer ofiary a docelowy zasób, do którego ofiara chce mieć dostęp. Jest kilka rodzajów tych ataków, ale wszystkie cechuje przechwytywanie wysyłanych komunikatów między dwiema stronami i możliwość podszywania się pod jedną ze stron. Ten atak kojarzy się przede wszystkim negatywnie przez wykorzystywanie go przez intruzów w sieci, lecz może być również zastosowany legalnie do monitorowania i analizy ruchu w sieci. W szczególności tutaj na myśl przychodzi zastosowanie przy testowaniu i analizie aplikacji webowych lub w środowisku pracy, gdzie administrator sieci, chciałby mieć kontrolę i wgląd w ruch odbywający się w jego sieci. Jednym z popularniejszych narzędzi jest mitmproxy (oprócz tego, możemy jeszcze wyróżnić podobne programy: Burp Suite, Sslsplit, Squid).

Mitmproxy

Mitmproxy - zestaw narzędzi, które zapewniają przechwytywanie zaszyfrowanego ruchu przez SSL/TLS przez proxy dla protokołów HTTP/1, HTTP/2 i Websocketów.

Do głównych funkcji zaliczamy:

- Przechwytywanie i modyfikacja na bieżąco zapytań i odpowiedzi HTTP i HTTPS.
- Zapisywanie zapytań
- Odtwarzanie ze strony klienta i serwera wcześniej zapisanych zapytań i odpowiedzi
- Tryb reverse-proxy by przekierowywać ruch od określonych serwerów
- Tryb transparent-proxy na systemach macOS i Linux
- Tworzenie rozszerzeń za pomocą języka Python i mitmproxy API
- Generowanie na bieżąco certyfikatów SSL/TLS
- [inne](#)

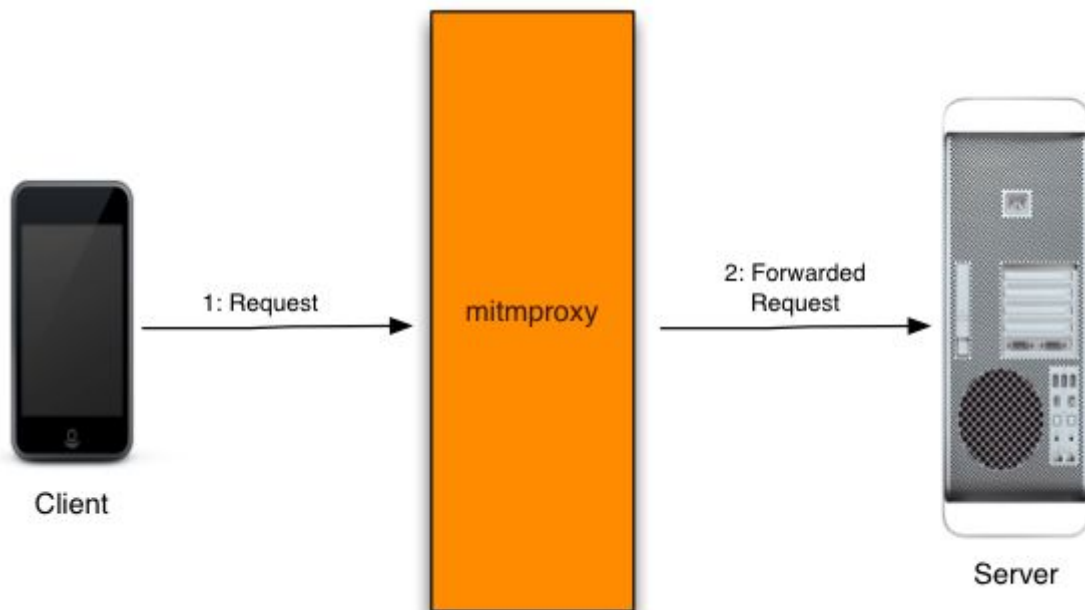
Mitmproxy zawiera 3 główne narzędzia: Po krótkce zapoznamy się z każdym w poszczególnych zadaniach.

- mitmproxy
- mitmweb
- mitmdump

W tym laboratorium przedstawimy najważniejsze funkcje mitmproxy, lecz po całość możliwości odsyłamy na oficjalną stronę <https://docs.mitmproxy.org>.

Jak mitmproxy działa?

Skonfigurowanie klienta do użycia proxy przy protokole HTTP nie jest skomplikowane i polega na prostym przekierowaniu zapytania przez proxy, tak jak przedstawiono to na poniższym rysunku.



Zupełnie inaczej wygląda sytuacja z protokołem HTTPS, zaszyfrowanym przez protokoły SSL/TLS.

Podczas połączenia do strony z protokołem https, klient wysła następujące zapytanie:

```
CONNECT example.com:443 HTTP/1.1
```

Tradycyjny proxy z powodu protokołu TLS/SSL nie może zobaczyć lub zmanipulować zaszyfrowanego zapytania, więc po prostu otwiera pipe pomiędzy klientem a serwerem. Proxy staje się jedynie pośrednikiem który może tylko przekierowywać dane zapytania bez możliwości zajrzenia do środka.

W celu ominięcia szyfrowania SSL/TLS, mitmproxy generuje na bieżąco, z każdą nową domeną do której wysyłają zapytanie klient, własne certyfikaty CA (Certificate Authority), lecz najpierw by zapewnić, że klient będzie ufał tym certyfikatom musimy ręcznie zarejestrować zaufany certyfikat CA. Będziemy to wykonywać w pierwszych zadaniach przy konfiguracji mitmproxy.

Aby generować na bieżąco certyfikaty potrzebna jest nazwa domenowa strony musimy poradzić sobie z 3 problemami.

- **Jaki jest hostname?**

W przypadku, gdy klient łączy się poprzez adres IP zapytanie wygląda następująco:

```
CONNECT 10.1.1.1:443 HTTP/1.1
```

Mitmproxy używa wtedy mechanizmu [upstream certificate sniffing](#). Gdy tylko pojawia się zapytanie CONNECT, zatrzymujemy część konwersacji klienta i w tym samym czasie podejmujemy osobne połączenie do serwera. Realizujemy TLS handshake z serwerem i wyciągamy z użytego certyfikatu parametr Common Name do wygenerowania certyfikatu dla klienta.

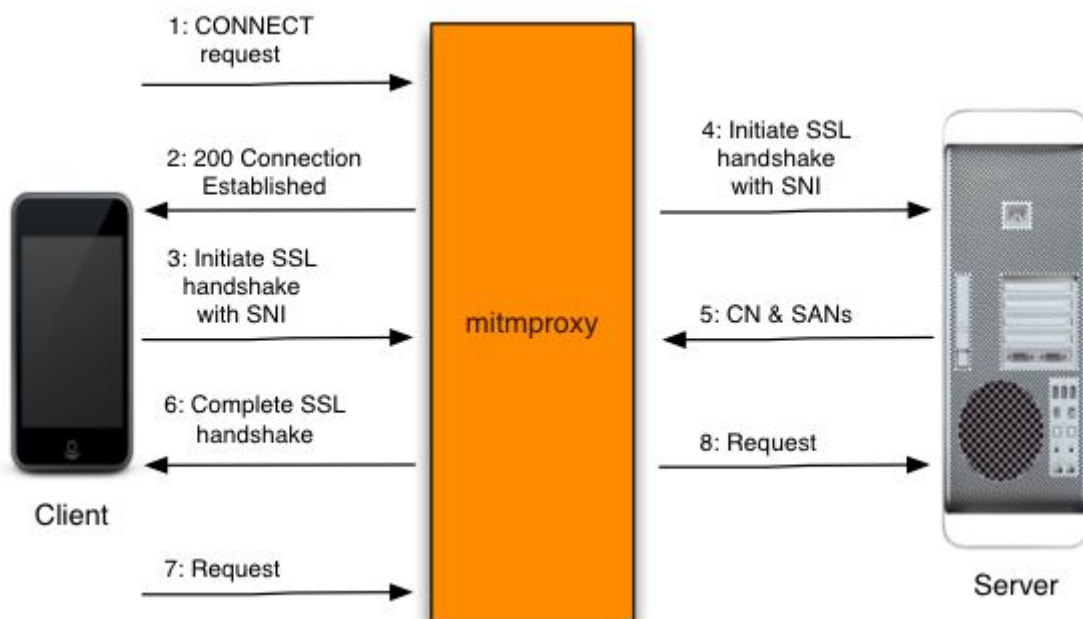
- **Subject Alternative Name**

Czasem Common Name certyfikatu nie jest tak naprawdę konkretnym hostname z którym się łączymy. Opcjonalne pole [Subject Alternative Name](#) pozwala na definiowanie alternatywnych domen. Rozwiązaniem jest wyciągnięcie pola SANs z certyfikatu otrzymanego z upstream certificate sniffing i dodanie go do wygenerowanego certyfikatu.

- **Server Name Indication**

Jednym z ograniczeń oryginalnego protokołu TLS jest to, że każdy certyfikat potrzebuje własnego adresu IP. Nie pozwala to na użycie takich mechanizmów jak [virtual hosting](#). Dlatego dodano rozszerzenie [Server Name Indication](#) do protokołu TLS. Pozwala to na zdefiniowanie remote server name na początku TLS handshake, co pozwala na wybranie prawidłowego certyfikatu na dokończenie procesu.

Po szczegółowe informacje odsyłamy na oficjalną stronę - <https://docs.mitmproxy.org/stable/concepts-howmitmproxyworks>.



HSTS - *HTTP Strict Transport Security*

Protokół HTTP nigdy nie był projektowany z myślą o bezpieczeństwie. W czasie, kiedy powstawały jego pierwsze wersje, z sieci komputerowych korzystały naukowe ośrodki obliczeniowe i paru naukowców. W aktualnej wersji 1.1 także nic w tym obszarze nie zmieniono. Wynikiem tego jest sytuacja, że do transmisji danych używany jest czysty tekst. Z tego powodu tak ważne jest używanie wersji HTTPS, czyli tunelowania protokołu HTTP w SSL/TLS. Nota bene zalecane jest używanie wersji TLS 1.1 lub 1.2, ponieważ starsze wersje posiadają liczne błędy. Szyfrowanie znacząco ogranicza możliwości atakującego w zakresie podglądania informacji o sesji lub jej modyfikacji. By to ominąć atakujący przekierowywali po prostu przeglądarkę użytkownika do komunikacji przez protokół HTTP. (W 2009r. powstał popularny program SSLStrip, który pozwalał na to jedną komendą).

Rozwiązaniem tych problemów jest nagłówek Strict-Transport-Security. Działa on tak, że jeżeli przeglądarka zobaczy, że witryna wysłała ten nagłówek, to przez czas określony w nagłówku cała komunikacja będzie się odbywać po HTTPS. Jest to działanie na poziomie przeglądarki, więc jeżeli użytkownik z niewiedzy lub przez roztargnienie będzie próbował połączyć się z wersją HTTP, to przeglądarka automatycznie podmieni jego zapytanie na HTTPS oraz zmieni wszystkie występujące na stronie linki na HTTPS.

```
Strict-Transport-Security: max-age=31536000; includeSubDomains
```

W tym przypadku, przeglądarka dostaje informacje, że ma stosować do komunikacji ze stroną, jak i z jej poddomenami, wyłącznie protokół HTTPS przez 31.536.000s.

Pojawia się jednak problem: użytkownik przy pierwszym połączeniu do serwera nie jest chroniony. Jego przeglądarka dopiero po zobaczeniu nagłówka wie, jak ma się dalej zachowywać. Pozostawia to lukę w bezpieczeństwie...

Dostawcy przeglądarek starają się rozwiązać ten problem przez zastosowanie tego samego podejścia jak przy certyfikatach CA. Istnieje wbudowana w program lista stron, dla których nagłówek HSTS jest już ustawiony.

[Źródło](#)

Wiedza o HSTS przyda nam się przy zadaniu 5. Na chwilę obecną ciężko znaleźć stronę, która nie posiada mechanizmu HSTS. Łatwo to sprawdzić - zmień na początku twojego url tekst z "https://nazwa_strony.pl" na "http://nazwa_strony.pl". Jeśli zmiana pozostała możesz rozwiązać zadanie EXTRA na dodatkowy punkt.

2. Instrukcja obsługi

Docker

W następujących ćwiczeniach, najczęściej będziemy korzystać z komendy:

```
$ docker run --rm -it -v ~/tmp/mitmproxy:/home/mitmproxy/.mitmproxy -p 8080:8080 mitmproxy/mitmproxy
```

Docker jest to zaawansowane narzędzie, posiadającą szeroki wachlarz funkcji. Nie będziemy ich tu wszystkich wyjaśniać. Przedstawimy tylko te, które będą używane przy wykonywaniu zadań z tego laboratorium.

Komenda uruchamiająca kontener ma następującą składnię:

```
$ docker run <flagi konfiguracyjne> <nazwa obrazu> <komendy przesyłane do kontenera>
```

Flagi konfiguracyjne:

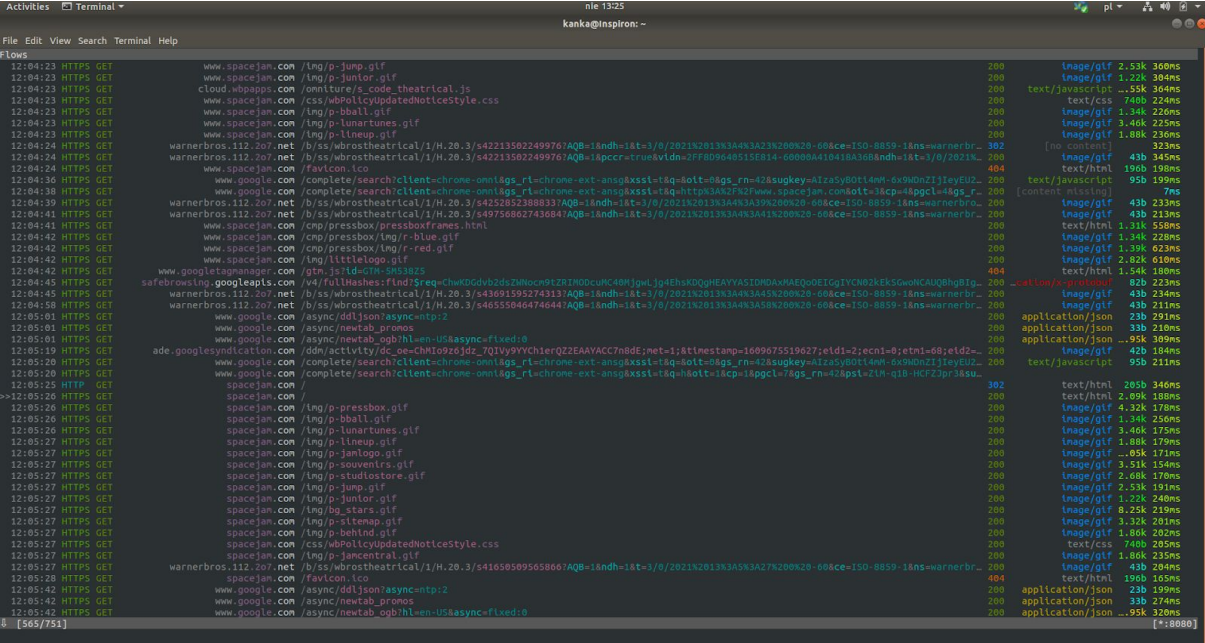
- p** <nr portu hosta>:<nr portu kontenera> - mapowanie portów
- v** <ścieżka do folderu na hoście>:<ścieżka do folderu na kontenerze> - Tworzenie wolumenów, które pozwalają na współdzielenie danych przez kontener i hosta. Zapobiegają również utracie danych przy każdym wyłączeniu kontenera.
- rm** - Automatyczne czyszczenie. Usuwa kontener po zamknięciu kontenera.
- it** - Pozwala na interakcję z terminalem poprzez terminal.

Mitmproxy - Ustawienie certyfikatu CA.

Przy pierwszym uruchomieniu mitmproxy w folderze /home/.mitmproxy zostaną wygenerowane certyfikaty CA. Aby się do nich dostać tworzymy volumen, który będzie odpowiadał folderowi /home/.mitmproxy w środku kontenera. W poleceniach w instrukcji domyślnie jest to miejsce /tmp/mitmproxy. Stamtąd należy wziąć certyfikat do zaimportowania.

Mitmproxy - Przechwytywanie zapytań

Jak szybko można zauważyć mitmproxy przechwytuje wszystkie zapytania http i https jakie wykonujesz. Każdy wiersz odpowiada jednemu przechwytywanemu zapytaniu. Możesz użyć mitmproxy do dokładnej analizy ruchu webowego, a dzięki zaimportowaniu certyfikatu CA, dane są w odszyfrowanej postaci.



```
nie 13:25
kanka@Inspiron: ~
File Edit View Search Terminal Help

Flows
12:04:23 HTTPS GET www.spacejam.com /img/p-jump.gif 200 image/gif 2.58k 300ms
12:04:23 HTTPS GET www.spacejam.com /img/p-junk.gif 200 image/gif 1.22k 304ms
12:04:23 HTTPS GET cloud.wpapps.com /omnture/s_code_theatrical.js 200 text/javascript ...55k 364ms
12:04:23 HTTPS GET www.spacejam.com /css/wbPolicyUpdatedNoticeStyle.css 200 text/css 740b 224ms
12:04:23 HTTPS GET www.spacejam.com /img/p-bball.gif 200 image/gif 1.34k 226ms
12:04:23 HTTPS GET www.spacejam.com /img/p-lunartunes.gif 200 image/gif 3.46k 225ms
12:04:23 HTTPS GET www.spacejam.com /img/p-l lineup.gif 200 image/gif 1.88k 236ms
12:04:24 HTTPS GET warnerbros.112.207.net /b/ss/wbrostheatrical/1/H.20.3/s42213502249976?AQ=1&ndh=1&t=3/8/2021&2013K3A4K3A45N200N20-68&ce=150-8859-1&ns=warnerbr... 302 [no content] 323ms
12:04:24 HTTPS GET warnerbros.112.207.net /b/ss/wbrostheatrical/1/H.20.3/s42213502249976?AQ=1&pcr=tr=ue&vldn=177809640515814-60080410418A36&ndh=1&t=3/8/2021N... 404 image/gif 43b 145ms
12:04:24 HTTPS GET www.spacejam.com /favicon.ico 200 text/html 196b 198ms
12:04:30 HTTPS GET www.google.com /complete/search?client=chrome-omnigags_rl=chrome-ext-ansg&xssi=t&q=htp&hl=en-US&asyncl=0 200 text/javascript 95b 199ms
12:04:30 HTTPS GET warnerbros.112.207.net /b/ss/wbrostheatrical/1/H.20.3/s4252852388833?AQ=1&ndh=1&t=3/8/2021&2013K3A4K3A394200N20-68&ce=150-8859-1&ns=warnerbro... 404 [content=1&id=0] 7ms
12:04:39 HTTPS GET warnerbros.112.207.net /b/ss/wbrostheatrical/1/H.20.3/s49758602743684?AQ=1&ndh=1&t=3/8/2021&2013K3A4K3A45N200N20-68&ce=150-8859-1&ns=warnerbro... 200 image/gif 43b 233ms
12:04:41 HTTPS GET www.spacejam.com /cmp/pressbox/pressboxframes.html 200 text/html 1.1k 250ms
12:04:42 HTTPS GET www.spacejam.com /cmp/pressbox/img/r-blue.gif 200 image/gif 1.34k 228ms
12:04:42 HTTPS GET www.spacejam.com /cmp/pressbox/img/r-red.gif 200 image/gif 1.39k 623ms
12:04:42 HTTPS GET www.spacejam.com /img/littlelogo.gif 200 image/gif 2.82k 610ms
12:04:42 HTTPS GET www.googletagmanager.com /gtm.js?id=cn-sss825 404 text/html 1.54k 188ms
12:04:45 HTTPS GET safelinks.ling.googleapis.com /v4/fullHashes?flnd7Sreq=ChwDGGvB2dsZMocn9LZINODCUC40WJgwLJg4EhskDQgHEAYVASIDMD&v=AEQo0EIGgIYCN02kEkSowNCAUQBhgBig... 200 application/javascript 82b 223ms
12:04:45 HTTPS GET warnerbros.112.207.net /b/ss/wbrostheatrical/1/H.20.3/s43691595274313?AQ=1&ndh=1&t=3/8/2021&2013K3A4K3A45N200N20-68&ce=150-8859-1&ns=warnerbr... 200 image/gif 43b 234ms
12:04:50 HTTPS GET warnerbros.112.207.net /b/ss/wbrostheatrical/1/H.20.3/s4655846474644?AQ=1&ndh=1&t=3/8/2021&2013K3A4K3A45N200N20-68&ce=150-8859-1&ns=warnerbr... 200 image/gif 43b 211ms
12:05:01 HTTPS GET www.google.com /async/ddljson?async=ntp:2 200 application/json 23b 291ms
12:05:01 HTTPS GET www.google.com /async/newtab_promos 200 application/json 33b 210ms
12:05:01 HTTPS GET www.google.com /async/newtab_ogb?hl=en-US&asyncl=0 200 application/json ...95k 109ms
12:05:19 HTTPS GET ade.googleadsyndication.com /dm/activity/dc-oe-CHM10926jdr_7Q1U9y9YchlerQZ2EAYVACC7n0dE?met=1&timestamp=1899675519627&eld=1&ecnl=0&etml=68&eld2=... 200 image/gif 42b 104ms
12:05:20 HTTPS GET www.google.com /complete/search?client=chrome-omnigags_rl=chrome-ext-ansg&xssi=t&q=hott&ikcp=1&pgcl=7&gs_rn=42&psl=ZLN-q1B-HCFZ3pr3&su... 200 text/javascript 95b 211ms
12:05:25 HTTP GET spacejam.com / 302 text/html 205b 346ms
12:05:26 HTTPS GET spacejam.com /img/p-pressbox.gif 200 image/gif 4.32k 178ms
12:05:26 HTTPS GET spacejam.com /img/p-bball.gif 200 image/gif 1.34k 256ms
12:05:26 HTTPS GET spacejam.com /img/p-lunartunes.gif 200 image/gif 3.46k 175ms
12:05:27 HTTPS GET spacejam.com /img/p-l lineup.gif 200 image/gif 1.88k 179ms
12:05:27 HTTPS GET spacejam.com /img/p-janlogo.gif 200 image/gif ...05k 171ms
12:05:27 HTTPS GET spacejam.com /img/p-souvenirs.gif 200 image/gif 3.51k 154ms
12:05:27 HTTPS GET spacejam.com /img/p-studiostore.gif 200 image/gif 2.68k 170ms
12:05:27 HTTPS GET spacejam.com /img/p-jump.gif 200 image/gif 2.53k 191ms
12:05:27 HTTPS GET spacejam.com /img/p-junk.gif 200 image/gif 1.22k 240ms
12:05:27 HTTPS GET spacejam.com /img/bg_stars.gif 200 image/gif 8.25k 219ms
12:05:27 HTTPS GET spacejam.com /img/p-sitemap.gif 200 image/gif 3.32k 201ms
12:05:27 HTTPS GET spacejam.com /css/wbPolicyUpdatedNoticeStyle.css 200 text/css 740b 205ms
12:05:27 HTTPS GET spacejam.com /img/p-jancentral.gif 200 image/gif 1.86k 235ms
12:05:27 HTTPS GET warnerbros.112.207.net /b/ss/wbrostheatrical/1/H.20.3/s4165858955866?AQ=1&ndh=1&t=3/8/2021&2013K3A5K3A27N200N20-68&ce=150-8859-1&ns=warnerbr... 200 image/gif 43b 204ms
12:05:28 HTTPS GET spacejam.com /favicon.ico 404 text/html 196b 165ms
12:05:42 HTTPS GET www.google.com /async/ddljson?async=ntp:2 200 application/json 23b 199ms
12:05:42 HTTPS GET www.google.com /async/newtab_promos 200 application/json 33b 274ms
12:05:42 HTTPS GET www.google.com /async/newtab_ogb?hl=en-US&asyncl=0 200 application/json ...95k 208ms
```

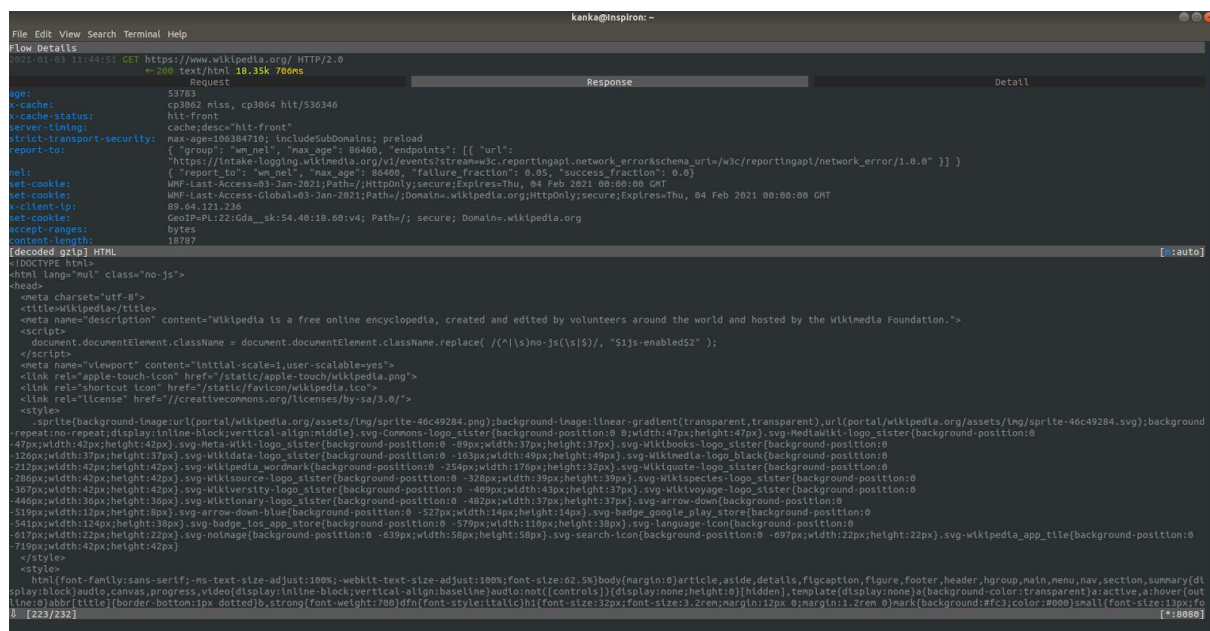
Rys 1. Panel mitmproxy.

Po kliknięciu w dowolny przechwycony komunikat, ukazać ci się do wyboru 3 panele:

Request - możesz zobaczyć tutaj zapytanie które zostało wysłane na daną stronę.

Response - odpowiedź serwera na zapytanie

Detail - szczegóły dotyczące serwera (adresy ip, certyfikaty CA...)



Rys 2. Przykładowe zdjęcie panelu po wybraniu przechwyconego zapytania.

Mitmproxy - Ustawienie filtrów

Po krótkim czasie ilość przechwyconych zapytań może być przytłaczająca, możemy wtedy ustawić filtry. Naciśnij 'f' lub wpisz ": set view_filter 'google.com' "" by znaleźć zapytania tylko dotyczące domeny google. Jest to najprostszy sposób filtrowania pakietów, by sprawdzić bardziej zaawansowane funkcję sprawdź: <https://docs.mitmproxy.org/stable/concepts-filters/>.

Mitmproxy - Zatrzymywanie zapytań

Przydatną funkcją mitmproxy jest zatrzymywanie zapytań w trakcie ruchu. Przechwycone zapytanie jest wtedy wstrzymywane, użytkownik może wtedy odrzucić lub zmodyfikować dane zapytanie przed wysłaniem do serwera. Przechwytywanie wszystkich zapytań nie jest wygodne, ponieważ przerywa ci to co chwilę poprawną pracę przeglądarki. Dlatego mitmproxy oczekuje na podanie filtru, jako argument do komendy set intercept, który sprecyzuje nam informacje o zapytaniu, który chcemy przejąć. Naciśnij 'i' lub wpisz ":set intercept """, po czym w cudzysłowie napisz filtr, by przechwycić stronę, która nas interesuje. Dla przykładu by przechwycić wszystkie strony, z domeny gov.pl, użyjemy komendy :set intercept '~u .*\.gov.pl'

Po złapaniu danej strony, zostanie ona podświetlona na czerwono w panelu mitmproxy i będzie czekała na dalsze operacje. Możemy wznowić ruch za pomocą zaznaczenia danej strony (strzałki >> muszą wskazywać na dane zapytanie) za pomocą przycisku 'a' lub odrzucić za pomocą 'X' (Uwaga: Duża litera X). Modyfikowanie pakietu opiszemy w kolejnym podpunkcie.

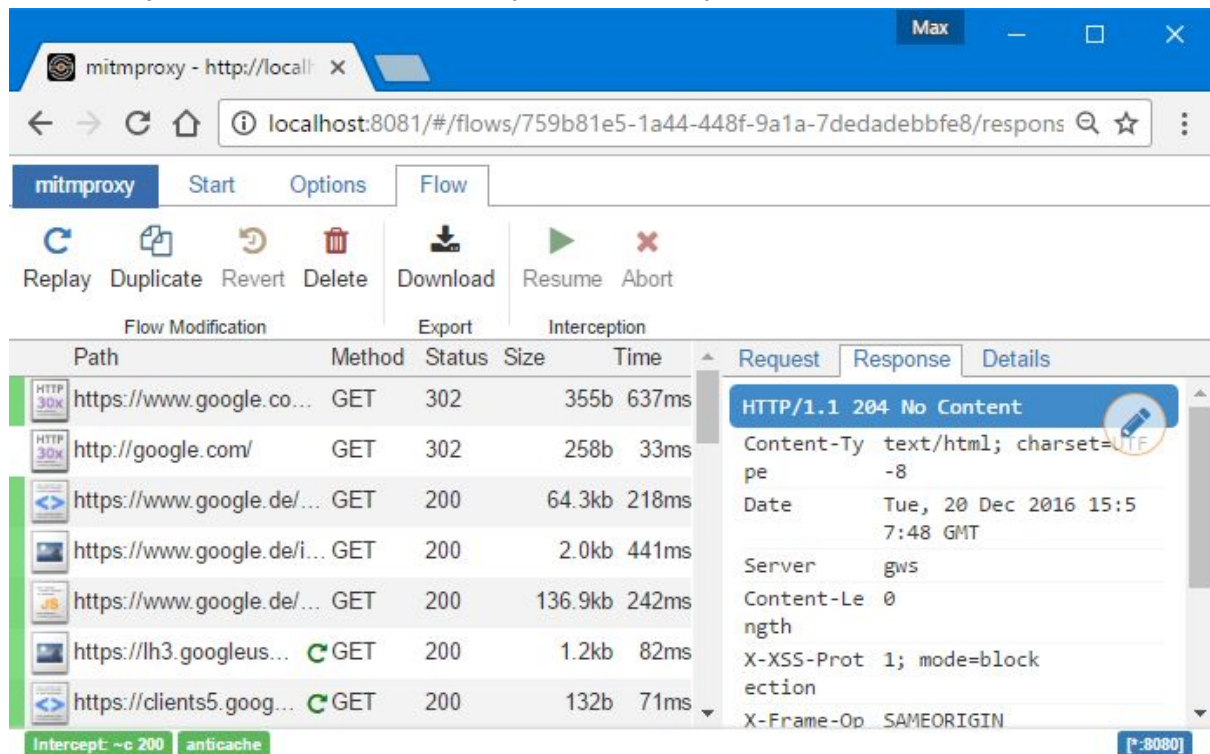
Aby usunąć filtr przechwytyjący, powtórz komendę zostawiając cudzysłów pusty (wpisz -> ":set intercept """).

Mitmproxy - Modyfikacja zapytań

Po przechwyceniu zapytania możemy modyfikować jego zawartość. Należy wybrać dane zapytanie (strzałki >> muszą wskazywać na zapytanie), a następnie nacisnąć przycisk 'e'. Ukaze się menu, z którego można wybrać którą część chcemy edytować.

Mitmweb

Mitmweb - jest to web-based GUI narzędzia mitmproxy.



Rys 3. Panel mitmweb.

Możemy go uruchomić za pomocą komendy:

```
$ docker run --rm -it -v ~/tmp/mitmproxy:/home/mitmproxy/.mitmproxy -p 8080:8080  
-p 8081:8081 mitmproxy/mitmproxy mitmweb --web-host 0.0.0.0
```

Uruchamiamy tutaj nie sam program mitmproxy lecz wersję przeglądarkową mitmweb. Samo proxy jest uruchomione na porcie 8080 localhosta, lecz podprogram strona mitmweb znajduje się na porcie 8081. Wejdź na stronę localhost:8081 lub 0.0.0.0:8081 aby ujrzeć interfejs mitmweb.

Na chwilę obecną mitmweb jest w wersji beta, więc wciąż brakuje mu dużo funkcji mitmproxy. Na potrzeby tego laboratorium można użyć mitmproxy jak i mitmweb.

Gdy wybierzemy z panelu opcję Start, ujrzymy tam miejsca na filtrowanie zapytań, wstrzymywanie ich lub podświetlanie. Mitmweb obsługuje tą samą składnię do tworzenia

filtrów jaka znajduje się w mitmproxy. (Mitmweb jest nadal w wersji beta, nie wszystkie filtry będą działać).

Mitmdump

Mitmdump udostępnia funkcjonalność podobną do narzędzia tcpdump. Pozwala na analizę, zapisywanie i automatyzację transformacji na ruchu HTTP.

Dzięki mitmdump możemy sami pisać potrzebne dla nas rozszerzenia w postaci skryptów w języku python wraz z użyciem mitmproxy API. Poprzez odpowiadanie na zdarzenia, które są generowane podczas pracy mitmproxy, możemy wchodzić w interakcje z zapytaniami i zmieniać sposób zachowania mitmproxy.

Uruchomić mitmdump możemy za pomocą komendy:

```
$ docker run --rm -it -v ~/tmp/mitmproxy:/home/mitmproxy/.mitmproxy -p 8080:8080  
mitmproxy/mitmproxy mitmdump <argumenty>
```

Aby użyć skryptów należy najpierw przesłać je do użycia przez kontener. Najlepiej użyć do tego volumen, który ustawiamy za pomocą flagi “-v”. Domyślnie w komendach volumen wskazujemy na ścieżkę ~/tmp/mitmproxy ze strony hosta, a /home/mitmproxy/.mitmproxy ze strony kontenera. To znaczy, że to co znajdzie się w folderze ~/tmp/mitmproxy na hoście, znajdziemy w kontenerze pod ścieżką /home/mitmproxy/.mitmproxy

Przykład

Ściągnijmy pliki potrzebne do realizacji laboratorium za pomocą narzędzia git:

```
$ git clone https://github.com/Kankarollo/MiTMInDocker.git
```

Przekopiuujemy skrypt anatomy.py z folderu addons do ścieżki ~/.mitmproxy

```
$ cp addons/anatomy.py ~/tmp/mitmproxy/
```

Od strony konteneru skrypt anatomy.py znajdzie się pod ścieżką /home/mitmproxy/.mitmproxy/anatomy.py, w takim razie taką też ścieżkę musimy podać podczas uruchomienia kontenera. Aby powiedzieć programowi mitmdump, żeby uruchomił skrypt, używamy flagi “-s”:

```
$ docker run --rm -it -v ~/tmp/mitmproxy:/home/mitmproxy/.mitmproxy -p 8080:8080  
mitmproxy/mitmproxy mitmdump -s /home/mitmproxy/.mitmproxy/anatomy.py
```

Skrypt ten ma za zadanie zliczać zapytania. W trakcie działania mitmdump powinniśmy obserwować wiadomości o treści “We’ve seen **X** flows”.

Analogicznie będziemy postępować w zadaniu 3.

Więcej o mitmdump sprawdź na stronie <https://docs.mitmproxy.org/stable/tools-mitmdump/>,
a o rozszerzeniach - <https://docs.mitmproxy.org/stable/addons-overview/>.

3. Zadania laboratoryjne

Uwagi dla rozwiązujących

- Do rozwiązania wszystkich zadań wymagany jest program Docker i przeglądarka internetowa (zalecamy przeglądarkę: Firefox ponieważ posiada opcje ustawienia osobistego proxy)
- System operacyjny - dowolny (Testowano na: Windows 10 i Ubuntu 18.04)
- W celu zapisania odpowiedzi na pytania proszę stworzyć plik tekstowy w dowolnej formie a na końcu przesłać na kurs przedmiotu. (Tam gdzie będzie można). Proszę podawać numery na które pytania podaje się odpowiedź.

0. Instalacja dockera

Do realizacji laboratoriów należy zainstalować program docker. W celu zainstalowania odpowiedniej wersji na poszczególne systemy operacyjne odsyłamy do oficjalnej strony: docker - [tutaj](#)

Pamiętaj by sprawdzić poprawność instalacji za pomocą komendy:

```
$ docker run hello-world
```

```
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
ca4f61b1923c: Pull complete
Digest:
sha256:ca0eeb6fb05351dfc8759c20733c91def84cb8007aa89a5bf606bc8b315b9fc7
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

1. Konfiguracja środowiska

1.1 Pobierz wymagane obrazy dockera:

```
$ docker pull mitmproxy/mitmproxy
```

1.2 Pobierz repozytorium.

Za pomocą narzędzia git pobierz repozytorium z dodatkowymi plikami potrzebnymi do realizacji zadań.

```
$ git clone https://github.com/Kankarollo/MiTMInDocker.git
```

Wejdź do repozytorium

```
$ cd MiTMInDocker
```

1.3 Uruchom mitmproxy

Uruchom mitmproxy za pomocą poniższej komendy.

Linux:

```
$ docker run --rm -it -v ~/tmp/mitmproxy:/home/mitmproxy/.mitmproxy -p 8080:8080 mitmproxy/mitmproxy
```

Windows:

```
$ docker run --rm -it -v <ścieżka do MiTMInDocker> /mitmproxy:/home/mitmproxy/.mitmproxy -p 8080:8080 mitmproxy/mitmproxy
```

Notka: Używając tej komendy utworzy nam się folder mitmproxy, który będzie volumenem naszego kontenera, po pierwszym uruchomieniu znajdziemy tam wygenerowane certyfikaty CA, które będziemy musieli zaimportować do przeglądarki w celu poprawnego działania programu. Pamiętaj by w każdej komendzie upewnić się, że wskazujemy na ten sam folder.

Powinna nam się ukazać konsola mitmproxy. Sprawdź czy reaguje na wysyłane żądanie za pomocą komend:

Linux:

```
$ http_proxy=http://localhost:8080/ curl http://example.com/  
$ https_proxy=http://localhost:8080/ curl -k https://example.com/
```

Windows:

<pomiń>

Na panelu mitmproxy powinniśmy zobaczyć 2 nowe wiersze, które zawierają informacje o zaobserwowanym zapytaniu HTTP do <http://example.com> i zapytaniu HTTPS do <https://example.com>.

1.4 Ustaw proxy.

Instrukcja dla przeglądarki Firefox:

- Wejdź w opcje -> Ogólne
- Na samym dole wejdź w zakładkę Sieć w Ustawienia...



- Ustaw proxy tak jak na zdjęciu:

A screenshot of the 'Konfiguracja proxy do łączenia z Internetem' (Proxy configuration for connecting to the Internet) dialog box in Firefox. The 'Ręczna konfiguracja serwerów proxy' (Manual proxy configuration) option is selected. The settings shown are: 'Serwer proxy HTTP' at 127.0.0.1 port 8080, with a checked box 'Użyj tego serwera proxy także dla FTP i HTTPS'; 'Serwer proxy HTTPS' at 127.0.0.1 port 8080; 'Serwer proxy FTP' at 127.0.0.1 port 8080; 'Host SOCKS' at port 0, with 'SOCKS v5' selected. There is also an option for 'Adres URL automatycznej konfiguracji proxy' which is currently empty. An 'Odśwież' (Refresh) button is at the bottom right.

W przypadku przeglądarki Chrome ustawienia proxy są pobierane z systemu. Dlatego trzeba ustawić proxy dla całego systemu operacyjnego. Instrukcja dla [Ubuntu](#), [Windows](#).

1: Spróbuj wejść na stronę <https://wp.pl> przez przeglądarkę internetową. Czy strona załaduje się bez żadnych kłopotów? Opisz co zauważyłeś? Przez jaki mechanizm jest to spowodowane?

1.4 Zaimportuj certyfikat CA.

Zaimportuj certyfikat CA w przeglądarce internetowej z której będziesz korzystał podczas wykonywania laboratorium.

Instrukcja dla przeglądarki Firefox:

- Wejdź w ustawienia w zakładkę Prywatność i bezpieczeństwo
- Znajdź podpunkt Certyfikaty i wybierz opcję "Wyświetl certyfikaty"

Certyfikaty

Kiedy serwer żąda osobistego certyfikatu użytkownika:

☐ wybierz certyfikat automatycznie

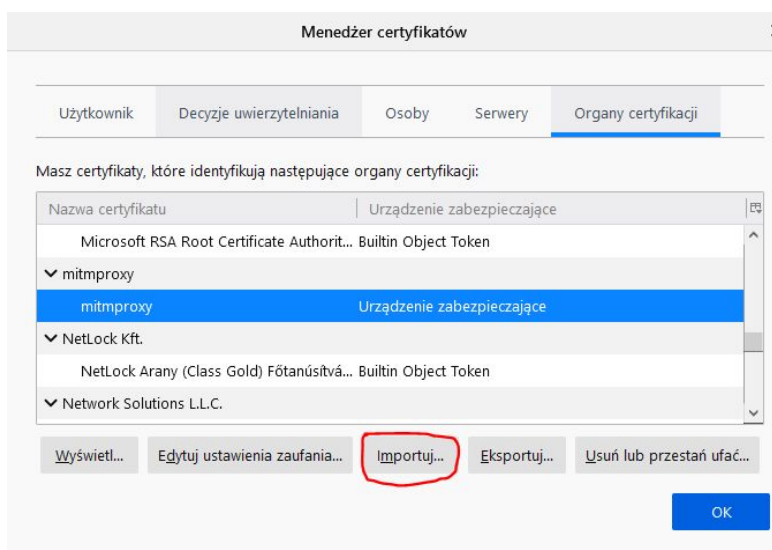
☒ pytaj za każdym razem

☒ Odpytywanie serwerów OCSP w celu potwierdzenia wiarygodności certyfikatów

[Wyświetl certyfikaty...](#)

[Urządzenia zabezpieczające...](#)

- Wybierz opcję Importuj i wybierz certyfikat. Powinieneś go znaleźć w folderze, który ustawiłeś jako twój volumen z kontenerem dockera. (Domyślnie ~/tmp/mitmproxy) Dla przeglądarki firefox, plik ten powinien nazywać się mitmproxy-ca-cert.pem.
- W przypadku poprawnego zaimportowania powinieneś móc znaleźć certyfikat na liście.



Instrukcja dla Chrome:

- Wejdź w ustawienia w zakładkę Prywatność i bezpieczeństwo
- Wejdź w pole Bezpieczeństwo
- Na samym dole wejdź w opcję "Zarządzaj certyfikatami"
- Wybierz opcję Importuj i wybierz certyfikat. Powinieneś go znaleźć w folderze, który ustawiłeś jako twój volumen z kontenerem dockera. (Domyślnie ~/tmp/mitmproxy) Dla systemu linux wybierz plik mitmproxy-ca-cert.pem. Dla systemu Windows wybierz plik mitmproxy-ca-cert.p12.
- W przypadku poprawnego zaimportowania powinieneś móc znaleźć certyfikat na liście.

Wejdź na dowolną stronę przez protokół https (np. <https://wp.pl>). W przypadku poprawnego skonfigurowania, strona powinna załadować się bez żadnych problemów.

2: Wejdź na stronę <https://wp.pl> i sprawdź w przeglądarce certyfikat, który jest używany(naciśnij na kłódkę przy pasku url i wejdź w szczegóły). Podaj nazwę certyfikatu. Wstaw zrzut ekranu panelu mitmproxy.

2. Analiza zapytań HTTP i HTTPS.

2.1 Filtracja zapytań w mitmproxy.

Po prawidłowym skonfigurowaniu proxy należy zapoznać się z funkcjami filtrowania i przechwytywania zapytań opisanych w drugim rozdziale i odpowiedzieć na poniższe pytania.

3: Wejdź na stronę internetową - [link](#), użyj funkcji filtrowania by na panelu były widoczne jedynie zapytania związane z podanym linkiem. Wstaw zrzut z ekranu panelu mitmproxy.

2.2 Uruchom mitmweb.

Mitmproxy zawiera również wersję GUI pod nazwą mitmweb. Więcej informacji o nim znajdziesz w drugim rozdziale.

Zakończ działanie mitmproxy i uruchom mitmweb poniższą komendą po czym otwórz jego interfejs wchodząc na adres <http://127.0.0.1:8081>. (Sprawdź czy flaga -v wskazuje na prawidłowy folder.)

Linux:

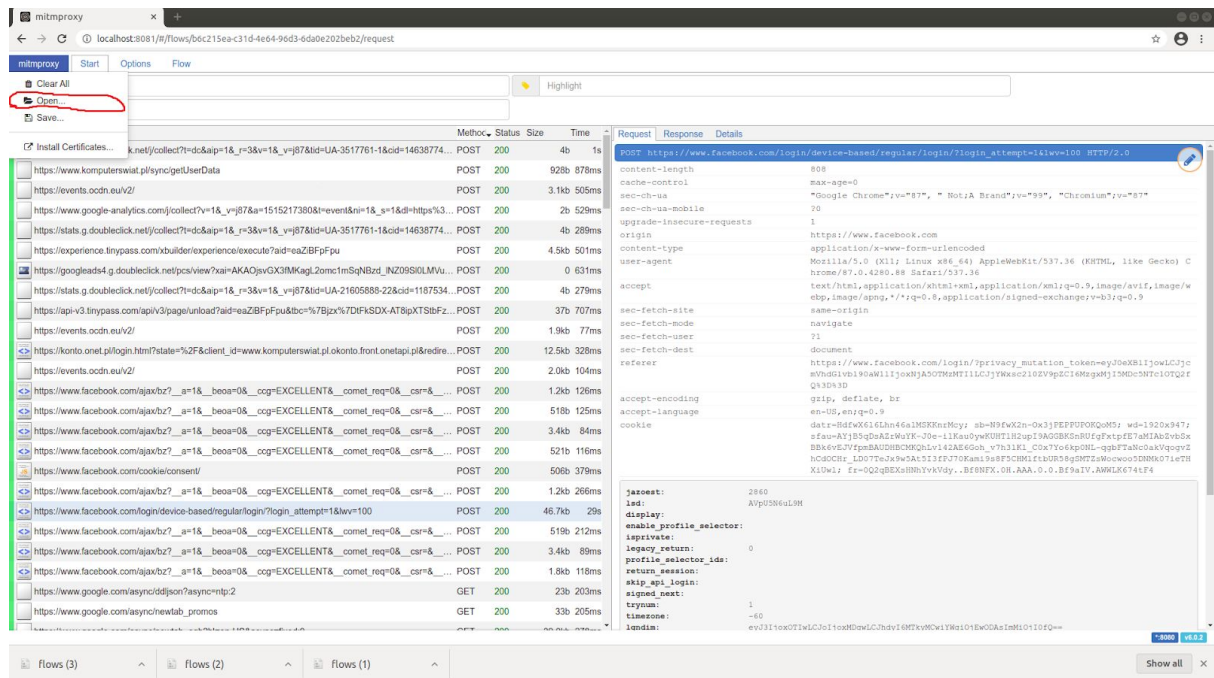
```
$ docker run --rm -it -v ~/tmp/mitmproxy:/home/mitmproxy/.mitmproxy -p 8080:8080  
-p 8081:8081 mitmproxy/mitmproxy mitmweb --web-host 0.0.0.0
```

Windows:

```
$ docker run --rm -it -v <ścieżka do MiTMInDocker>  
/mitmproxy:/home/mitmproxy/.mitmproxy -p 8080:8080 -p 8081:8081  
mitmproxy/mitmproxy mitmweb --web-host 0.0.0.0
```

W celu zapoznania się z narzędziem sprawdzimy jak poszczególne strony internetowe przesyłają dane logowania.

Otwórz plik "mitmweb_exercise/zadanie_znajdz_haslo" z poziomu mitmweb. Rozwiń pasek 'mitmproxy' z panelu w lewym górnym rogu i wybierz opcję open.



Załadują ci się przygotowane wcześniej przechwycone zapytania HTTP i HTTPS. Znajdują się tam zapytania, które mitmweb przechwycił podczas logowania się na strony komputerświat i facebook. Znajdź je do rozwiązania kolejnego zadania.

4: Znajdź zapytania które dotyczą logowania na stronie komputerświat. Wyciągnij z nich login i hasło admina i napisz je w odpowiedzi. Zrób to samo dla logowania w przypadku platformy facebook.com. W przypadku facebook opisz w jakiej formie widzisz hasło logowania.

(Podpowiedź: Szukaj po słowie 'login' w nazwie i po metodzie POST).

Kolejne zadanie można wykonać za pomocą dowolnego narzędzia. Pamiętaj, żeby zakończyć działanie poprzedniego narzędzia przed uruchomieniem kolejnego.

Informacje o HSTS znajdziesz we wprowadzaniu.

5: Za pomocą poznanych narzędzi sprawdź nagłówki strony internetowej [link](http://spacejam.com). Czy możemy rozpoznać mechanizm HSTS po nagłówkach na tej stronie? Uzasadnij odpowiedź. Spróbuj wejść na stronę <http://spacejam.com>. Co się stało?

(EXTRA) Znajdź stronę (poza tą z poprzedniego pytania) na której nie występuje mechanizm HSTS. Podaj url poniżej.

3. Modyfikowanie zapytań HTTP i HTTPS

3.1 Modyfikowanie odpowiedzi strony

W tym zadaniu ukażemy jedno z potężniejszych narzędzi mitmproxy czyli tworzenie rozszerzeń za pomocą mitmproxy API i języka python by automatyzować operacje na

zapytaniach HTTP i HTTPS. Do tego zadania użyjemy program mitmdump i przygotowane wcześniej skrypty w języku python. Mitmdump możemy uruchomić za pomocą poniższej komendy. (Sprawdź czy flaga -v wskazuje na prawidłowy folder.)

Linux:

```
$ docker run --rm -it -v ~/tmp/mitmproxy:/home/mitmproxy/.mitmproxy -p 8080:8080 mitmproxy/mitmproxy mitmdump <argumenty>
```

Windows:

```
$ docker run --rm -it -v <ścieżka do MiTMInDocker> /mitmproxy:/home/mitmproxy/.mitmproxy -p 8080:8080 mitmproxy/mitmproxy mitmdump <argumenty>
```

Uruchom skrypt block_urls.py z folderu addons, opieraj się na przykładzie znajdującym się pod koniec rozdziału 2 w ramach omawiania narzędzia mitmdump. W celu poprawnego działania programu pamiętaj o przekopiowaniu plików block_urls.py oraz urls.txt z folderu addons do volumena, który będzie współdzielony z kontenerem mitmproxy (domyślnie ~/tmp/mitmproxy/).

Wejdź na stronę <https://wp.pl> by sprawdzić działanie skryptu. Powinno ukazać ci się zdjęcie z napisem "Access Denied" oznaczające blokadę strony.

By pokazać prostotę tworzonych skryptów, otwórz skrypt w dowolnym edytorze tekstu. Znajdź w kodzie pole, w którym jako string jest zapisany kod HTML, który będziesz otrzymywał w odpowiedzi w przypadku wejścia na stronę internetową, która jest obecna na liście urls.txt.

6. Zmodyfikuj skrypt block_urls.py tak by wyświetlał dowolne inne zdjęcie (lub po prostu inną treść) w przypadku wejścia na stronę znajdującą się na blackliście -urls.txt. Wstaw zrzut ekranu przeglądarki po uruchomieniu twojej wersji skryptu.

3.2 Przekierowanie adresu

W tym przykładzie skorzystamy ze skryptu http_redirect.py do przekierowania adresów z naszej listy urls.txt na adres, który sami zechcemy.

Analogicznie do poprzedniego przykładu uruchom przykładowy skrypt http_redirect.py za pomocą mitmdump z poziomu dockera. I wejdź na stronę <https://wp.pl>. W przypadku gdy oryginalny cel podróży stosuje protokół HTTP/2, a strona na którą przekierowujemy nie wspiera tego protokołu, może dojść do błędu. W takich przypadkach należy dodać flagę --no-http2 na koniec komendy uruchamiającej mitmdump.

W przypadku poprawnego uruchomienia, po wejściu na stronę <https://wp.pl> powinniśmy zauważyć, że tak na prawdę znajdujemy się na stronie mitmproxy.org.

Gdy otworzymy skrypt w dowolnym edytorze tekstu, łatwo zobaczyć która linia kodu odpowiada za wybranie strony na którą przekierowujemy użytkownika.

7. Zmodyfikuj skrypt `http_redirect.py` tak by przekierowywał stronę z listy `urls.txt` na dowolną wybraną przez siebie stronę (inną niż w przykładzie). Wstaw zrzut ekranu przeglądarki po uruchomienia twojej wersji skryptu.

Po więcej przykładów odsyłam na oficjalną stronę mitmproxy
<https://docs.mitmproxy.org/stable/addons-examples/#http-redirect-requests>.

4. Czyszczenie (opcjonalne)

W celu usunięcia obrazu dockera użyj komend:

```
$ docker image ls
```

Znajdź wiersz który dotyczy obrazu mitmproxy/mitmproxy zapamiętaj jego TAG. A następnie usuń go komendą:

```
$ docker rmi <image_TAG>
```

4. Zakończenie

Jest to koniec zadań z tego laboratorium. Wyślij stworzony dokument tekstowy i nie zapomnij napisać też opinii co do zadań.

Dziękujemy za uwagę, mamy nadzieję, że się czegoś nowego nauczyliście! :)

P.S W przypadku wystąpienia pytań czy problemów zapraszamy do kontaktu jakąkolwiek drogą do któregoś z nas.