Project

Man In The Middle Attacks: Execution and Detection

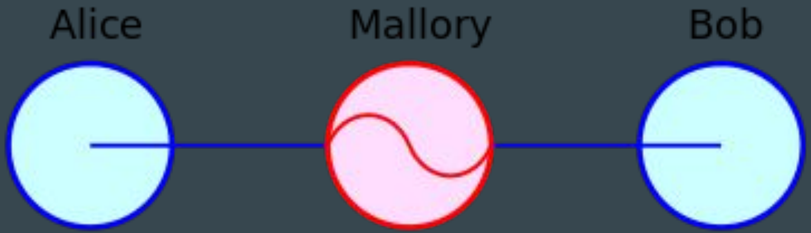
Daria Vaskovskaya

Agenda

- What is MITM?
- Project goals
- Work done during the iterations
- Future plan and current issues

MITM

Man-in-the-middle attack (MITM) is an attack where the attacker secretly relays and possibly alters the communications between two parties who believe that they are directly communicating with each other



Initial goal:

- Study ways of execution and detection of Man In The Middle Attacks.
- Set up a simulation platform
- Run various scripts that demonstrate performing such attacks, detecting them and circumventing that detection.

How to detect Mallory AND **be as silent as possible as Mallory to bypass those detections**

Iteration I

Goal:

- Study different variants of simulation platforms
- Set up the environment for Man In The Middle Attacks
- Perform some MITM attacks on simulation platforms

Study different variants of simulation platforms

Issues

• Lack of knowledge

• Lack of documentation

Choices:

- VirtualBox - flexibility, my experience with it, and community support.
- mininet - good documentation and simplicity

Study different variants of simulation platforms

Final choice:

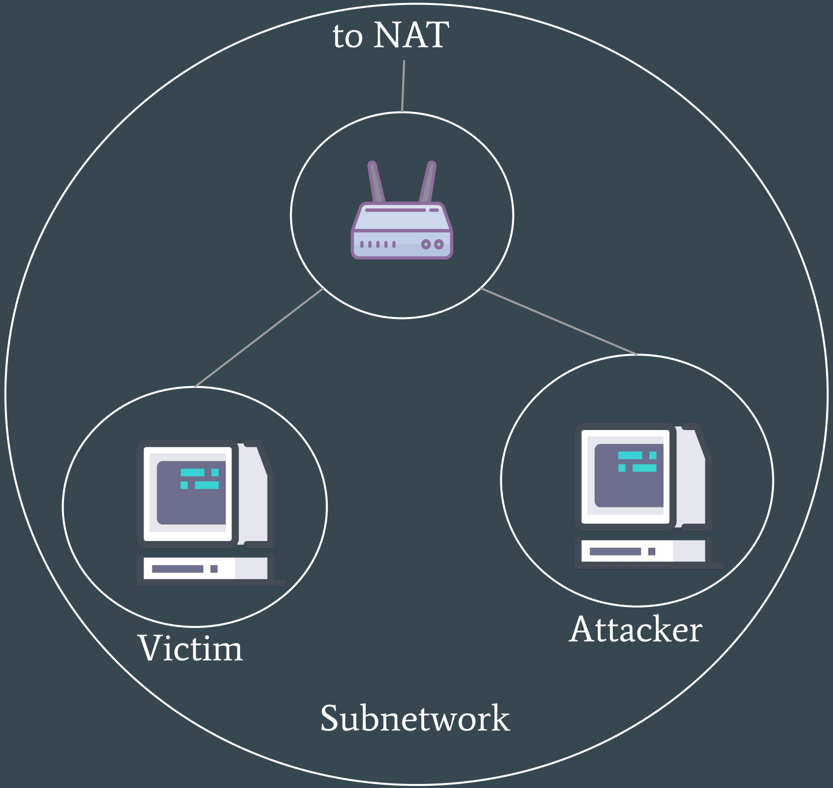
- Start with **VirtualBox** for its capabilities (because of not enough information about where the project will lead) and fast start-up (because of my experience).

- Then, **if mininet** (or other good out-of-the-box solution which is controlled by scripts) **will be capable of all the required configurations** for the showcase at the end of a project.

Set up the environment

Set-up instructions and scripts:
github

- Quagga - a network routing software suite
- DHCP



I Iteration Results

MITM Framework choice - Bettercap

Why bettercap?

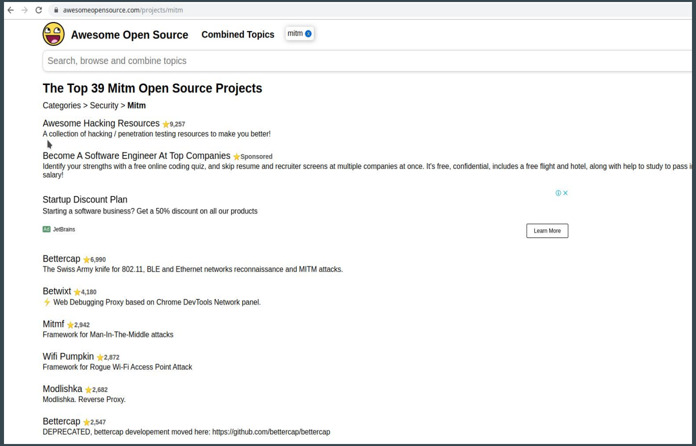
Based on:

- Most recent commit
- Stars
- Number of features

Based on info got from:

https://awesomeopensource.com/projects/mitm

My table with comparison:
Google Doc



awesomeopensource.com/projects/mitm

Awesome Open Source | Combined Topics | mitm

Search, browse and combine topics

The Top 39 Mitm Open Source Projects

Categories > Security > **Mitm**

Awesome Hacking Resources ⭐9,257
A collection of hacking / penetration testing resources to make you better!

Become A Software Engineer At Top Companies ⭐Sponsored
Identify your strengths with a free online coding quiz, and skip resume and recruiter screens at multiple companies at once. It's free, confidential, includes a free flight and hotel, along with help to study to pass i
salary!

Startup Discount Plan
Starting a software business? Get a 50% discount on all our products
Ad JetBrains
Learn More

Bettercap ⭐6,990
The Swiss Army knife for 802.11, BLE and Ethernet networks reconnaissance and MITM attacks.

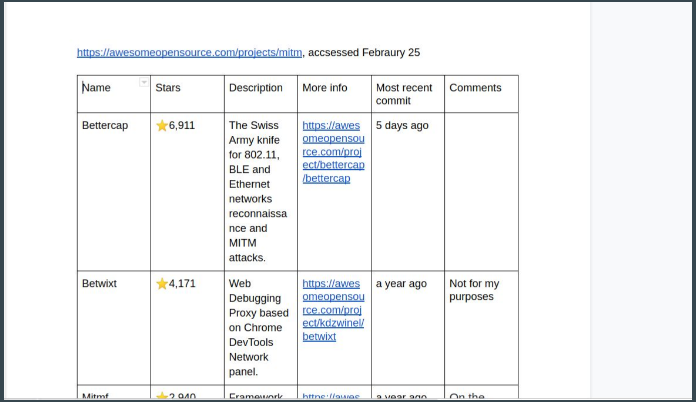
Betwixt ⭐4,180
⚡ Web Debugging Proxy based on Chrome DevTools Network panel.

Mitmf ⭐2,942
Framework for Man-In-The-Middle attacks

Wifi Pumpkin ⭐2,872
Framework for Rogue Wi-Fi Access Point Attack

Modlishka ⭐2,682
Modlishka. Reverse Proxy.

Bettercap ⭐2,547
DEPRECATED, bettercap developement moved here: https://github.com/bettercap/bettercap

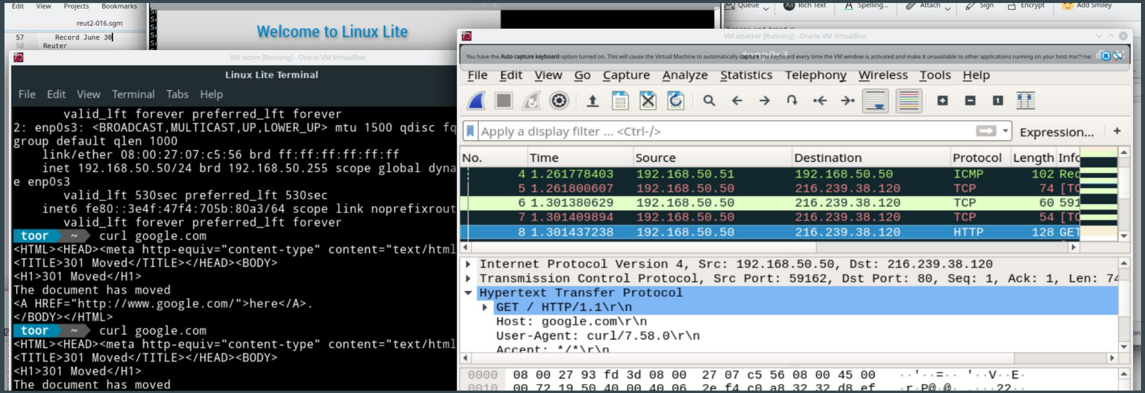


https://awesomeopensource.com/projects/mitm, accsessed Febraury 25

Name	Stars	Description	More info	Most recent commit	Comments
Bettercap	⭐6,911	The Swiss Army knife for 802.11, BLE and Ethernet networks reconnaissance and MITM attacks.	https://awesomeopensource.com/project/bettercap/bettercap	5 days ago	
Betwixt	⭐4,171	Web Debugging Proxy based on Chrome DevTools Network panel.	https://awesomeopensource.com/project/kdzwinel/betwixt	a year ago	Not for my purposes
Mitmf	⭐2,940	Framework	https://awes	a year ago	On the

Perform some MITM attacks on simulation platforms

ARP-spoofing



Linux Lite Terminal

```
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq
group default qlen 1000
    link/ether 08:00:27:07:c5:56 brd ff:ff:ff:ff:ff:ff
    inet 192.168.50.50/24 brd 192.168.50.255 scope global dyna
e enp0s3
       valid_lft 530sec preferred_lft 530sec
    inet6 fe80::3e4f:47f4:705b:80a3/64 scope link noprefixrout
       valid_lft forever preferred_lft forever
 toor  ~  curl google.com
<HTML><HEAD><meta http-equiv="content-type" content="text/html
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
<A HREF="http://www.google.com/">here</A>.
</BODY></HTML>
 toor  ~  curl google.com
<HTML><HEAD><meta http-equiv="content-type" content="text/html
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
```

```
No.  Time          Source          Destination      Protocol  Length Info
  4  1.261778403   192.168.50.51   192.168.50.50    ICMP         102 Rec
  5  1.261800607   192.168.50.50   216.239.38.120   TCP           74 [TC
  6  1.301380629   192.168.50.50   216.239.38.120   TCP           60 591
  7  1.301409894   192.168.50.50   216.239.38.120   TCP           54 [TC
  8  1.301437238   192.168.50.50   216.239.38.120   HTTP         128 GET
```

```
▸ Internet Protocol Version 4, Src: 192.168.50.50, Dst: 216.239.38.120
▸ Transmission Control Protocol, Src Port: 59162, Dst Port: 80, Seq: 1, Ack: 1, Len: 74
▾ Hypertext Transfer Protocol
  ▸ GET / HTTP/1.1\r\n
    Host: google.com\r\n
    User-Agent: curl/7.58.0\r\n
    Accept: */*\r\n
```

```
0000  08 00 27 93 fd 3d 08 00  27 07 c5 56 08 00 45 00   ··'··=·· '··V··E·
0010  00 72 19 50 40 00 40 06  2e f4 c0 a8 32 32 d8 ef   ·r·P@·@· ····22··
```

Welcome to Linux Lite
File Edit View Terminal Tabs Help
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

II Iteration

Decision:

- Detect the presence of MITM attack **regardless of the mechanism used to launch it**

Goals:

- Read the papers about MITM detection
- Make short summaries about the noteworthy ones
- Pick paper/papers to investigate further during the 3rd iteration

II Iteration results

Noteworthy papers:

- Vesper: Using Echo-Analysis to Detect Man-in-the-Middle Attacks in LANs [
- The Security Impact of HTTPS Interception
- Detection of MITM Attack in LAN Environment using Payload Matching
- Client-Side Web Proxy Detection from Unprivileged Mobile Devices

Selected:

"The Security Impact of HTTPS Interception"

- Can not detect proxies which closely mimic the request of the client
- Can add this feature to proxy to bypass detection

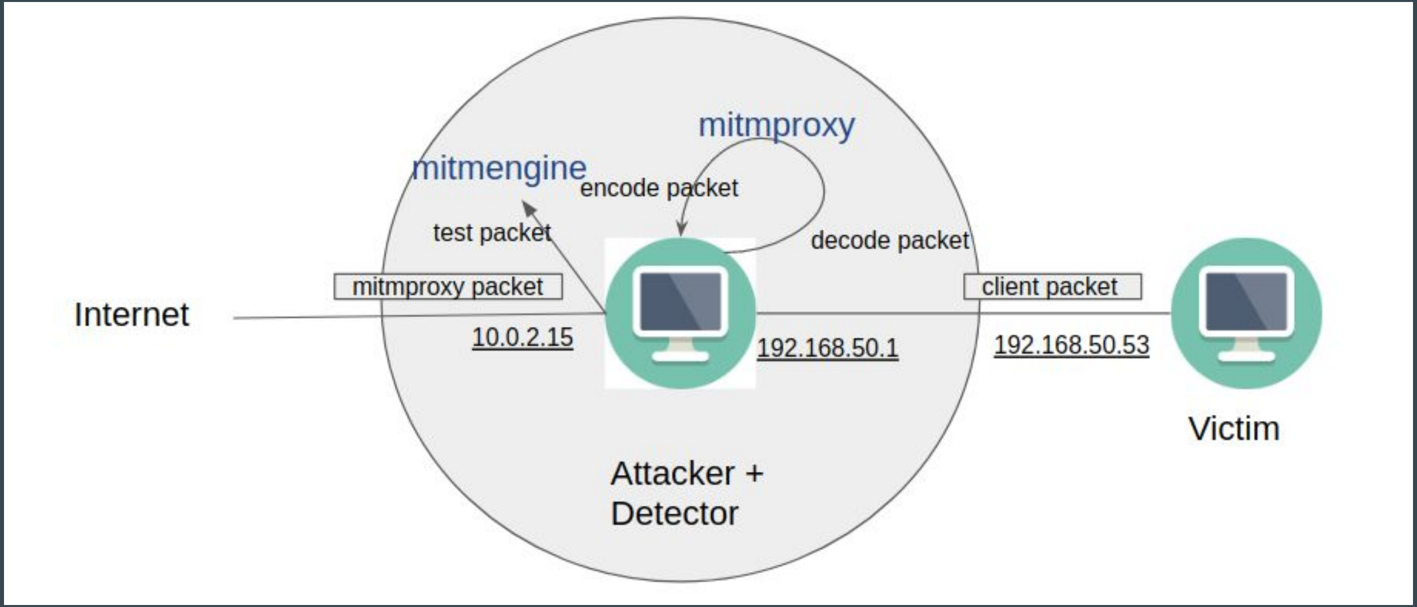
III Iteration

Goal:

Test and experiment with detection tool based on "The Security Impact of HTTPS Interception" paper - mitmengine

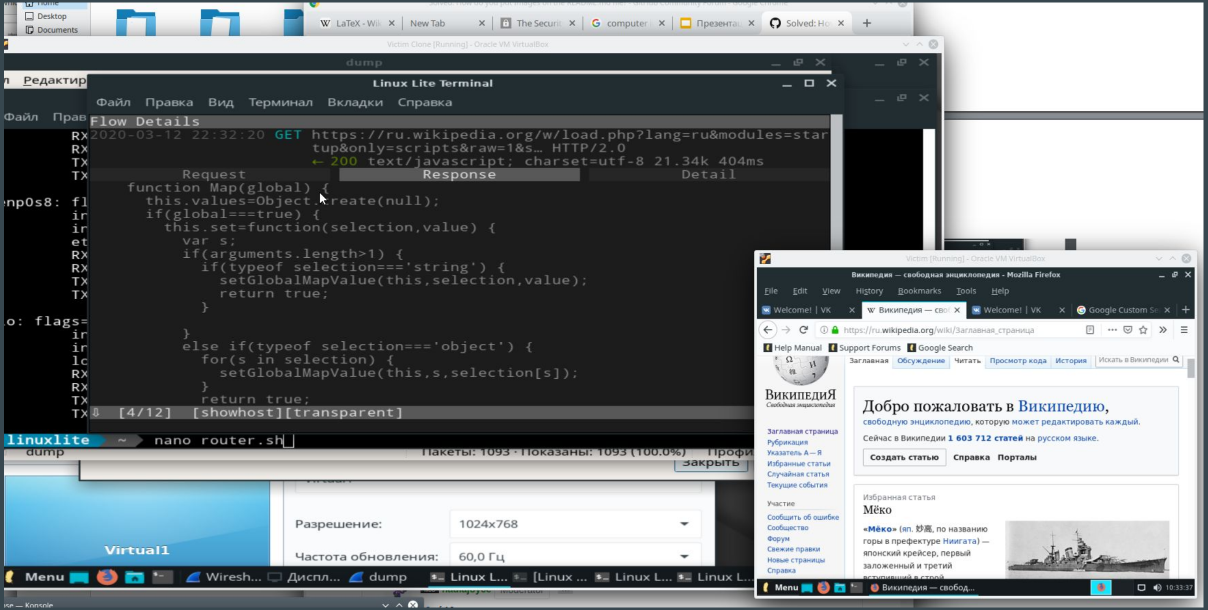
Results:

- Changed the set-up



Results

- Installed and ran mitmproxy to sniff HTTPs packets



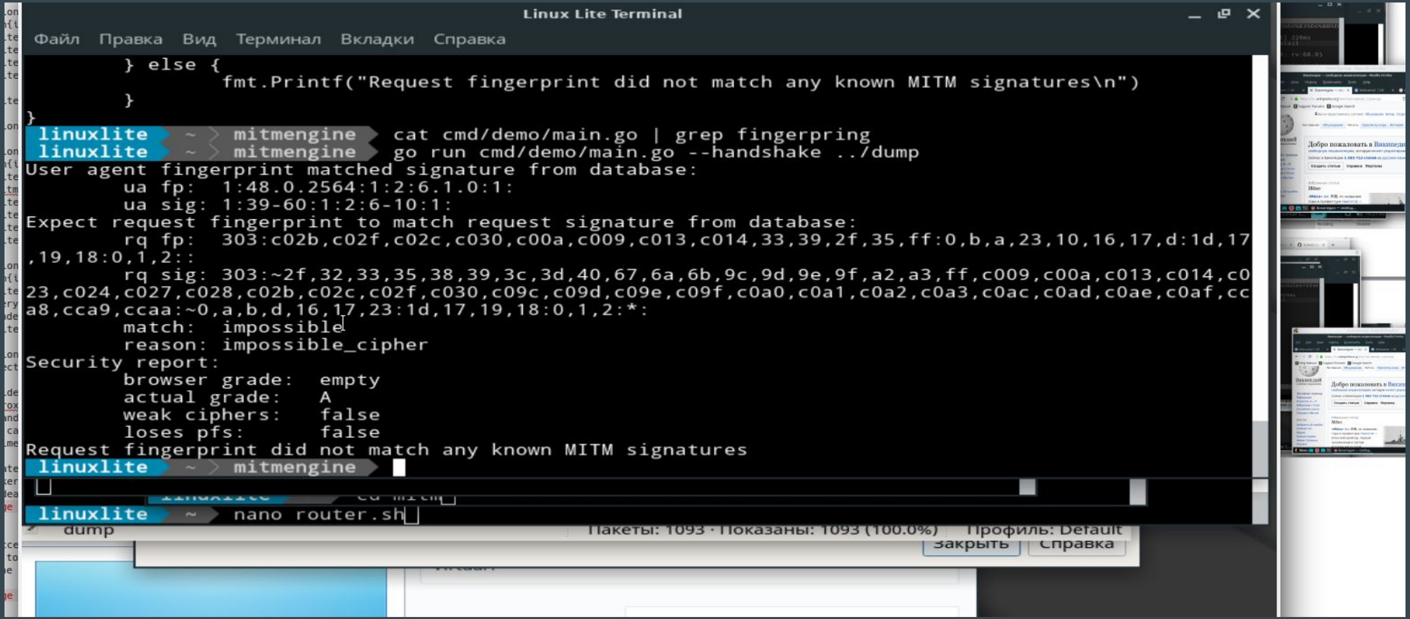
```
Flow Details
2020-03-12 22:32:20 GET https://ru.wikipedia.org/w/load.php?lang=ru&modules=star
                        tup&only=scripts&raw=1&s… HTTP/2.0
                        ← 200 text/javascript; charset=utf-8 21.34k 404ms
         Request                  Response                  Detail
  function Map(global) {
    this.values=Object.create(null);
    if(global===true) {
      this.set=function(selection,value) {
        var s;
        if(arguments.length>1) {
          if(typeof selection==='string') {
            setGlobalMapValue(this,selection,value);
            return true;
          }

        }
        else if(typeof selection==='object') {
          for(s in selection) {
            setGlobalMapValue(this,s,selection[s]);
          }
          return true;
⇩  [4/12]  [showhost][transparent]
linuxlite  ~  nano router.sh
```

Добро пожаловать в Википедию,
свободную энциклопедию, которую может редактировать каждый.
Сейчас в Википедии 1 603 712 статей на русском языке.

Избранная статья
Мёко
«Мёко» (яп. 妙高, по названию горы в префектуре Ниигата) — японский крейсер, первый заложенный и третий вступивший в строй

Results

- Installed detection tool in order to catch man in the middle
- Ran tests



```
        } else {
                fmt.Printf("Request fingerprint did not match any known MITM signatures\n")
        }
}
 linuxlite  ~  mitmengine  cat cmd/demo/main.go | grep fingerpring
 linuxlite  ~  mitmengine  go run cmd/demo/main.go --handshake ../dump
User agent fingerprint matched signature from database:
        ua fp:  1:48.0.2564:1:2:6.1.0:1:
        ua sig: 1:39-60:1:2:6-10:1:
Expect request fingerprint to match request signature from database:
        rq fp:  303:c02b,c02f,c02c,c030,c00a,c009,c013,c014,33,39,2f,35,ff:0,b,a,23,10,16,17,d:1d,17
,19,18:0,1,2::
        rq sig: 303:~2f,32,33,35,38,39,3c,3d,40,67,6a,6b,9c,9d,9e,9f,a2,a3,ff,c009,c00a,c013,c014,c0
23,c024,c027,c028,c02b,c02c,c02f,c030,c09c,c09d,c09e,c09f,c0a0,c0a1,c0a2,c0a3,c0ac,c0ad,c0ae,c0af,cc
a8,cca9,ccaa:~0,a,b,d,16,17,23:1d,17,19,18:0,1,2:*:
        match:  impossible
        reason: impossible_cipher
Security report:
        browser grade:  empty
        actual grade:   A
        weak ciphers:   false
        loses pfs:      false
Request fingerprint did not match any known MITM signatures
 linuxlite  ~  mitmengine 

 linuxlite  ~  nano router.sh
```

Issues and future plans

Issues:

Need more time to determine what happens exactly and why mitmproxy is not catched

Plans:

Dig into the code of mitmproxy and mitmengine in order to determine why mitmproxy is not detected.