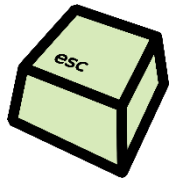
SOS_Proxy

Invisible Proxying Automation

Who Am I?



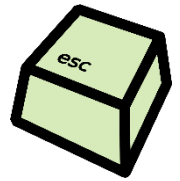
Lorenzo Comi

- IT Security Consultant @ Minded Security
- Focused on WebApp&Mobile Penetration Test
- Python lover -> NOT a professional/serious DEVELOPER… But I like to automate boring stuff
- Coffee addicted

MINDED SECURITY
The Software Security Company

https://me.me/embed/i/9d18af78efae44849d65f96030b1b23d

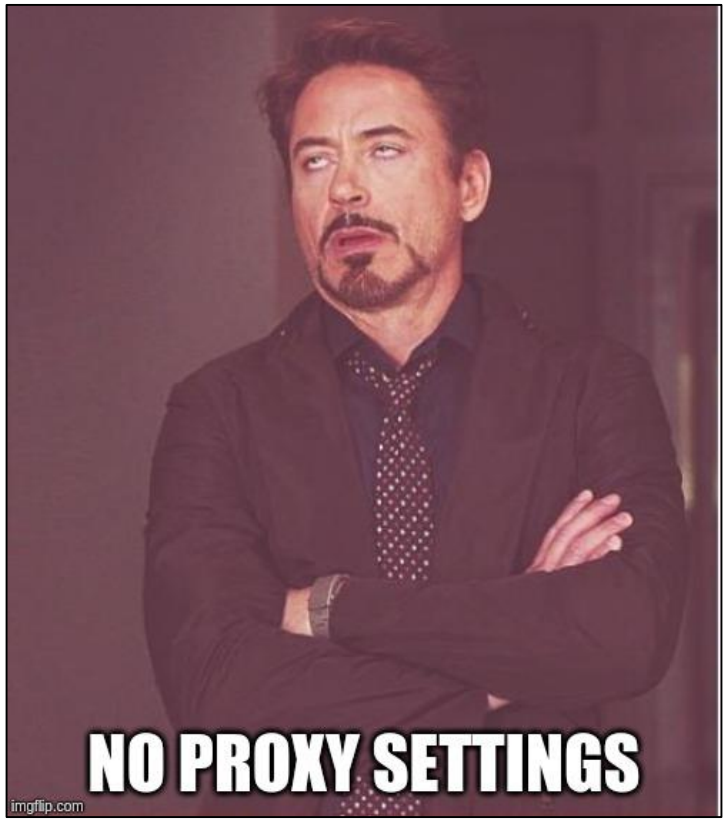
Why am I here?



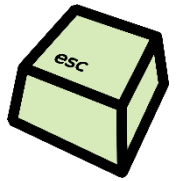
- Explain a technique to intercept HTTP traffic of a NON-proxy aware device
- Introduce SOS_Proxy tool to automate and scale this technique

The Problem

Is it possible to proxy a device that does not support this functionality?



Invisible proxy technique

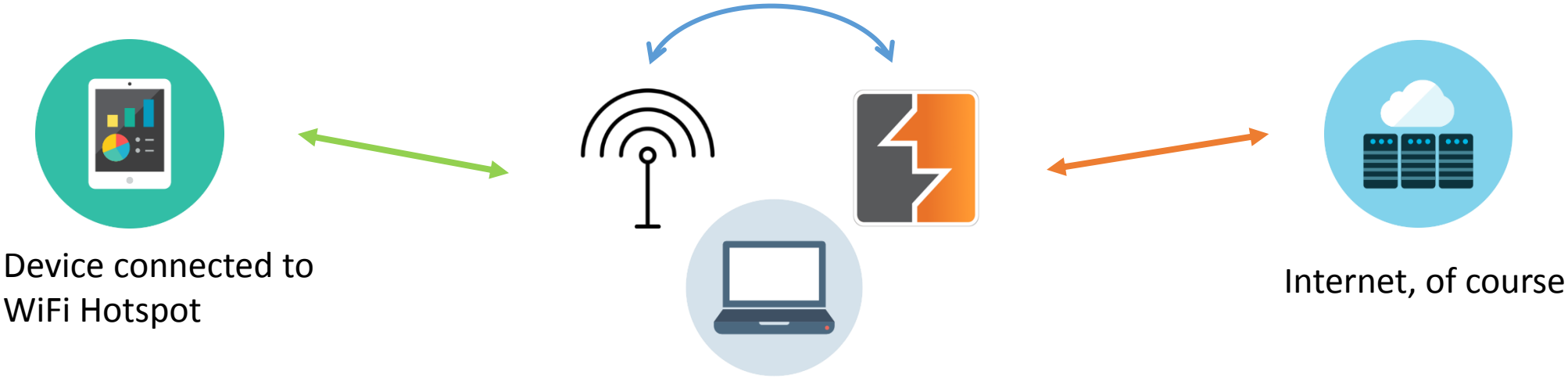


“Burp's support for invisible proxying allows non-proxy-aware clients to connect directly to a Proxy listener. […] Often, these clients don't support HTTP proxies, or don't provide an easy way to configure them to use one.”

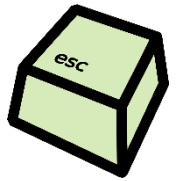
https://portswigger.net/burp/documentation/desktop/tools/proxy/options/invisible

Invisible proxy technique with one domain

Example: we want to intercept the HTTP traffic made by a specific app that send requests only to a specific domain.

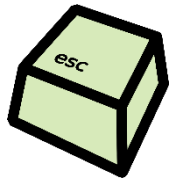


How it works



1. Intercept device DNS request in order to know which domains is calling.
2. For each domain create a new virtual interface.
3. Alter the hostname resolution through your hosts file.
4. Create a separate Proxy listener for each interface/domain.
5. Start intercepting!

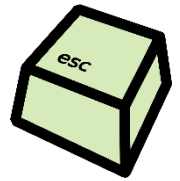
Invisible proxy technique with one domain



1. Mitm the device & log DNS traffic (es. With tcpdump)

```
$ sudo tcpdump -n -i wlan0 udp dst port 53 and src 10.42.0.228 -l
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wlan0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:42:21.928193 IP 10.42.0.228.51456 > 10.42.0.1.53: 13894+ A? googleads.g.doubleclick.net. (45)
12:42:22.788217 IP 10.42.0.228.18212 > 10.42.0.1.53: 2757+ A? pagead2.googlesyndication.com. (47)
12:42:28.400256 IP 10.42.0.228.52262 > 10.42.0.1.53: 37110+ A? secure-it.imrworldwide.com. (44)
12:42:28.427984 IP 10.42.0.228.37159 > 10.42.0.1.53: 27312+ A? www.corriere.it. (33)
12:42:29.697983 IP 10.42.0.228.52981 > 10.42.0.1.53: 30840+ A? images2.corriereobjects.it. (44)
```

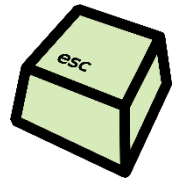
Invisible proxy technique with one domain



2. Create a virtual interface for any domain that you want to intercept

```
$ sudo ifconfig eth0:1 100.100.100.1
$ ifconfig | grep eth0 -A1
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.178.104  netmask 255.255.255.0  broadcast 192.168.178.255
--
eth0:1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 100.100.100.1  netmask 255.0.0.0  broadcast 100.255.255.255
```

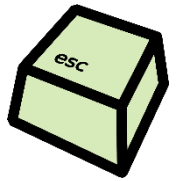
Invisible proxy technique with one domain



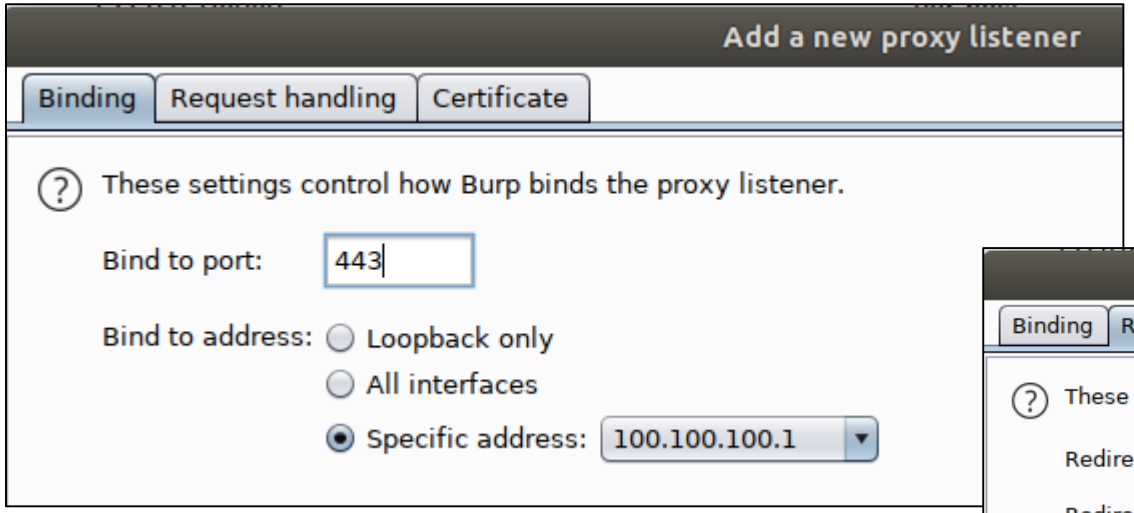
3. Perform a nslookup of the domain and set www.corriere.it resolution on your /etc/hosts config file

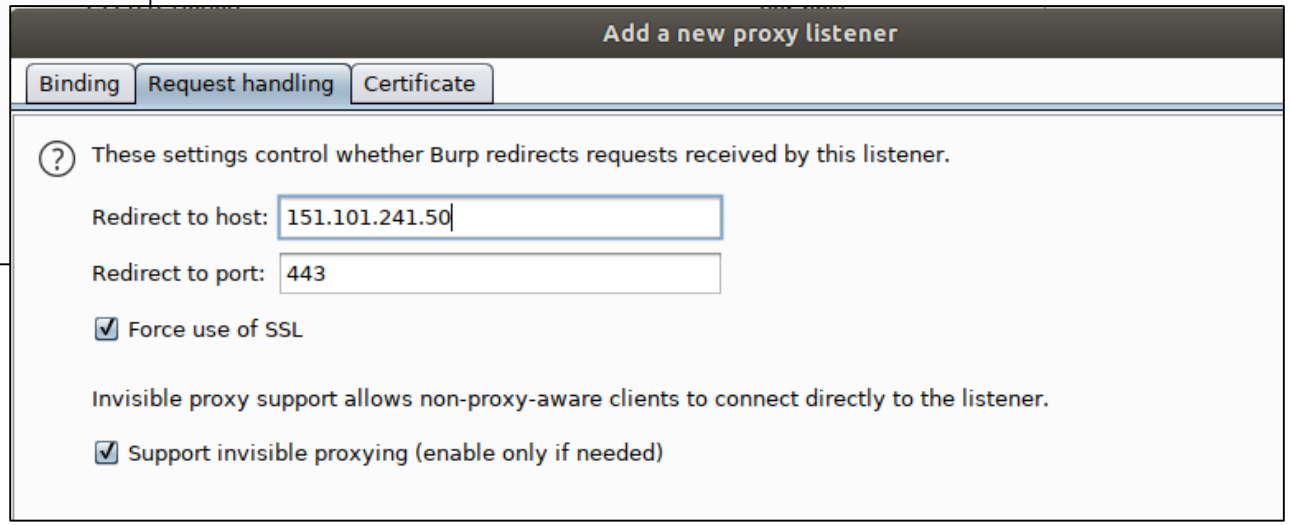
```
$ nslookup www.corriere.it | grep Address | grep -v '#'
Address: 151.101.241.50

$ echo '100.100.100.1    www.corriere.it' >> /etc/hosts
$ tail –n1 /etc/hosts
100.100.100.1    www.corriere.it
```

Invisible proxy technique with one domain

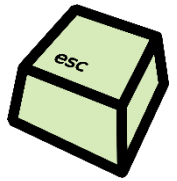


4. Set Burp proxy to redirect the traffic to the correct host





Invisible proxy technique with one domain



5. Start Intercepting!

Burp Project Intruder Repeater Window Help

Project options | User options | Deserialization Scanner | JSON Beautifier | Logger++ | Telewreck

Dashboard | Target | Proxy | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender

Intercept | HTTP history | WebSockets history | Options

Filter: Hiding CSS, image and general binary content

#	Host	Method	URL
3	https://151.101.241.50	GET	/apw.js?v=4&f=mod
2	https://151.101.241.50	GET	/apw.js?v=4
1	https://151.101.241.50	GET	/

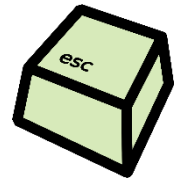
Request | Response

Raw | Params | Headers | Hex

```
GET / HTTP/1.1
Host: www.corriere.it
Connection: close
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
User-Agent: Mozilla/5.0 (Linux; Android 5.1.1; SAMSUNG SM-J320FN Build/LMY47V) AppleWebKit/537.36 (KHTML, like
Gecko) SamsungBrowser/3.5 Chrome/38.0.2125.102 Mobile Safari/537.36
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8
Cookie: apw_cache=0..1567700333.qWbRuA.IT.0...0.0...H7hJa1m5lZrcMu73hmRfmUOogOjZCNUAcuGG9pFbMwM; _cb_ls=1;
TSstop=NA|1567700337733; apw_browser=2131317223860227143.; cpmt_xa=5334; __ric=5334; CMP_ref=5334;
_hjid=8b4f66ef-a3c9-4f25-9390-96cb5f309378; rccsLocalPref=milano%7CMilano%7C015146;
rcsddfglr=1570295900.6.1.6JG8yUZY.2131317223860227143..EVKL_tmG-IeZR1Dgp2jK4k6ZUo8Sd4_901JpGjEZ-1U.81372d9e;
incognitoMode=false; apw_status=false; s_cc=true; s_fbsr=1; s_nr=1567760718417-Repeat; SC_LNK_CR=%5B%5BB%5D%5D;
gpv_pn=COR%2Fsport%2F; s_sq=%5B%5BB%5D%5D; s_ppvl=COR%2Fsport%2F%2C10%2C10%2C511%2C360%2C511%2C360%2C640%2C2%2CL;
_fbp=fb.1.1567702447427.858587868; s_ppv=COR%2Fsport%2F%2C10%2C10%2C511%2C360%2C511%2C360%2C640%2C2%2CL;
testcookie=true; _cb=CxPtsyBo6ffYCDtqte;
_chartbeat2=.1567700339300.1567760923561.11.BYjDMkBhrmRMC-l4BoC5SS7kClgssv.3; _cb_svref=null;
_ga=GA1.2.1723908148.1567700341; _gid=GA1.2.1367786737.1567700341
```

< | + | > | Type a search term | 0 matches

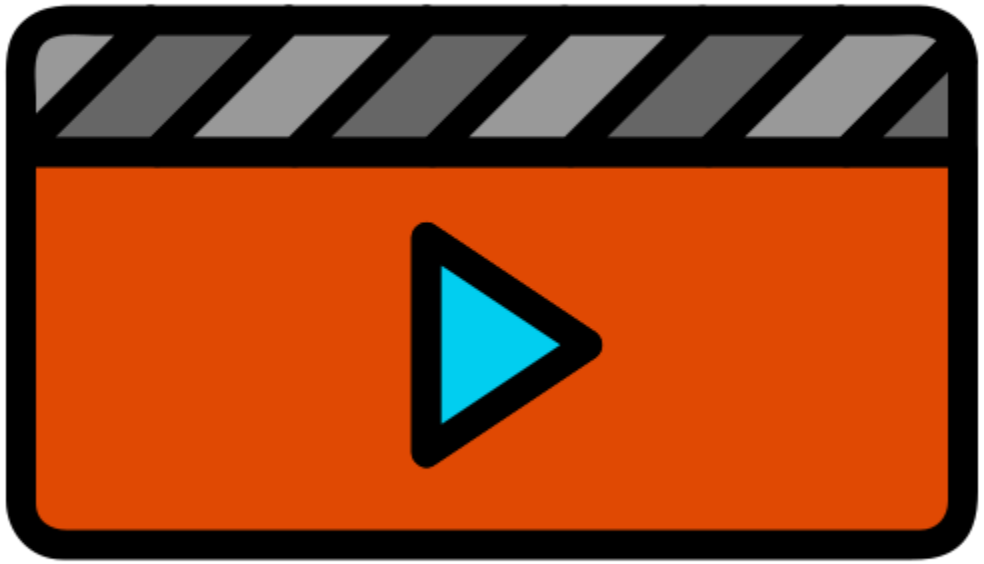
Automation and Scaling with SOS_Proxy



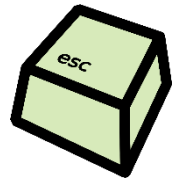
SOS_Proxy is a simple Python tool that automates the invisible proxy technique with the following features:

- DNS traffic sniffing == Domain_monitor
- Virtual interfaces creator
- Print information to set Burp's proxies
- Possibility to choose which domain has to be intercepted
- Possibility to backup and restore a hosts file configuration

Demo Time



SOS_Proxy possible improvements/ideas



1. Burp extension to automate proxies setup
2. DNS Poisoning?!
3. Python3
4. <Your_Idea_HERE>

Questions?

Thank You!

- https://github.com/c0mix/SOS_Proxy
- https://c0mix.github.io/