FIELD NOTE / MALWARE ANALYSIS

Malware Analysis Lab in ESXi: Isolated Network

Petruknisme · 3 Mar, 2023 · 4 min to read

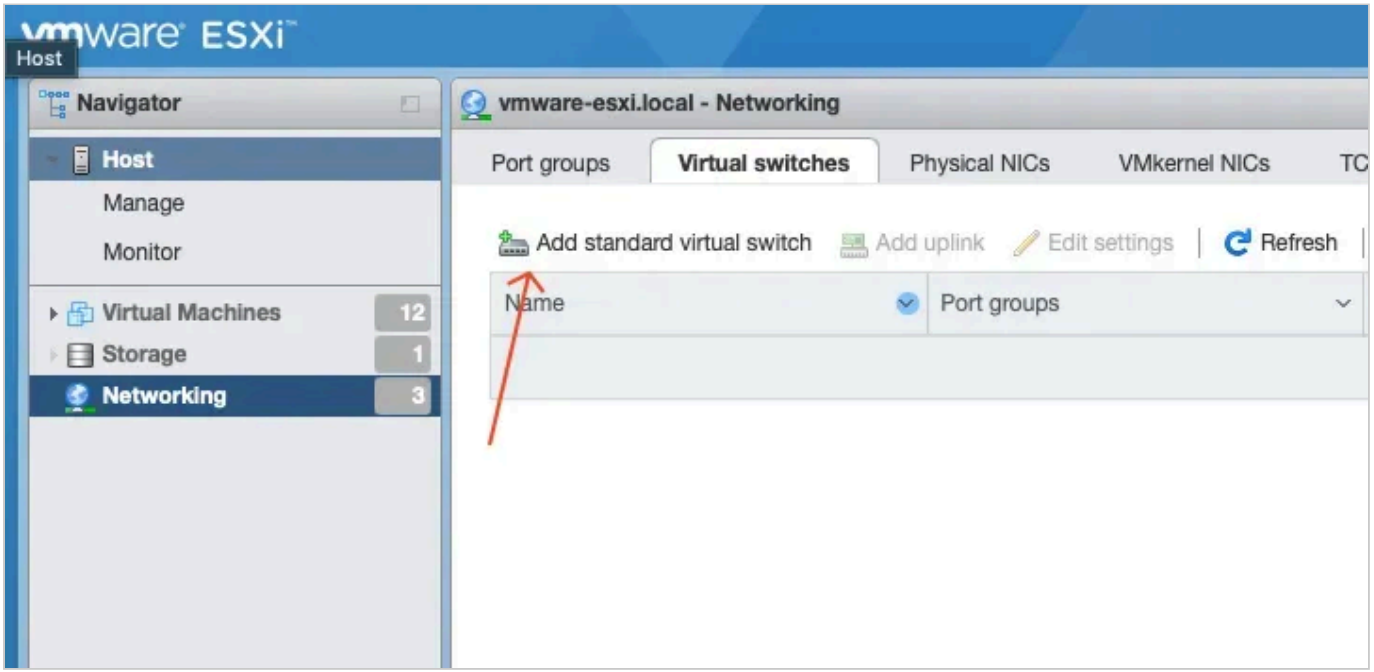
As you may be aware, performing malware analysis requires a secure and isolated network environment. This is because malware can potentially spread and infect other systems on the network, causing damage and compromising sensitive data.

Maybe you are already familiar with tutorial that demonstrate how to setup isolated network environment using vmware or virtualbox. In this tutorial, I don't want to use my own laptop for the malware analysis vm, it's too dangerous. Something like Guest-to-Host escape is possible, example https://secret.club/2021/01/14/vbox-escape.html (https://secret.club/2021/01/14/vbox-escape.html). It's better to use ESXi or something that not related to personal machine.

Similar to VirtualBox or VMware, the concept of "Host Only" networking can be used in ESXi to create an isolated network environment. However, unlike VirtualBox or VMware, ESXi does not have a pre-configured "Host Only" adapter network. As a result, it is necessary to manually configure a network adapter to create an isolated network environment.

Creating Host Only Adapter

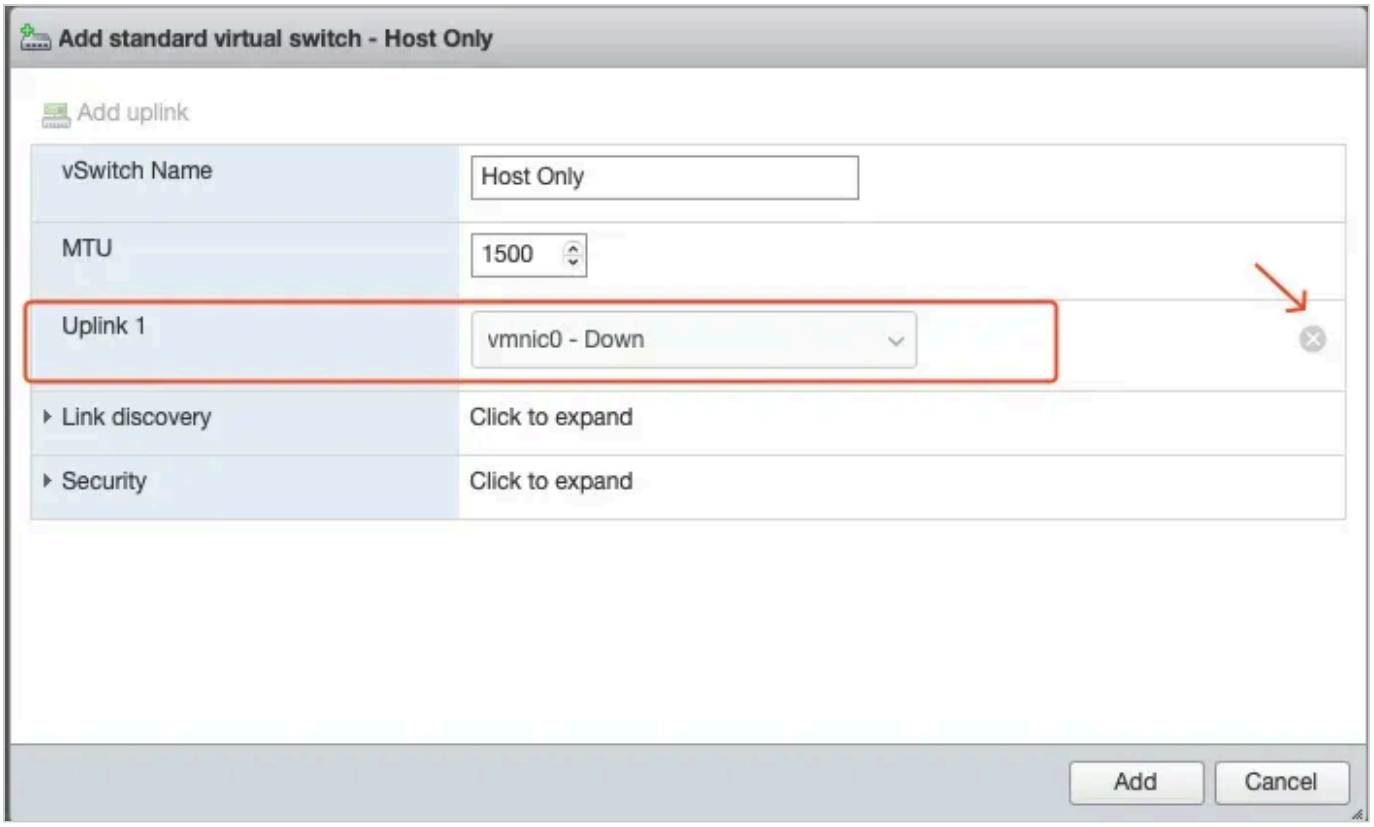
First we need to create vSwitch



Licensed under CC BY-SA 4.0

Add virtual switch

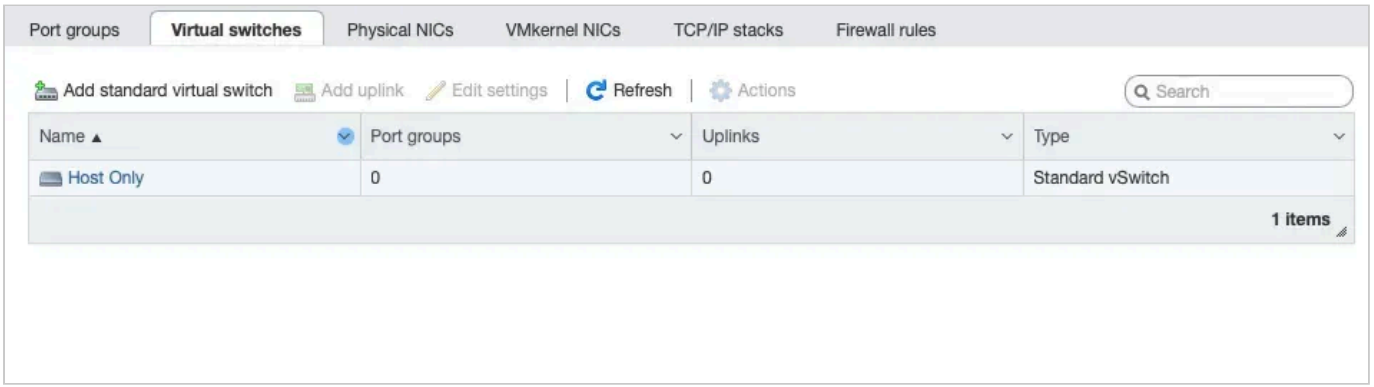
Click Add standard virtual switch



Add virtual switch and remove the uplink

Don't forget to **remove uplink** because we don't need uplink for the host only network. I repeat, **NO UPLINK**.

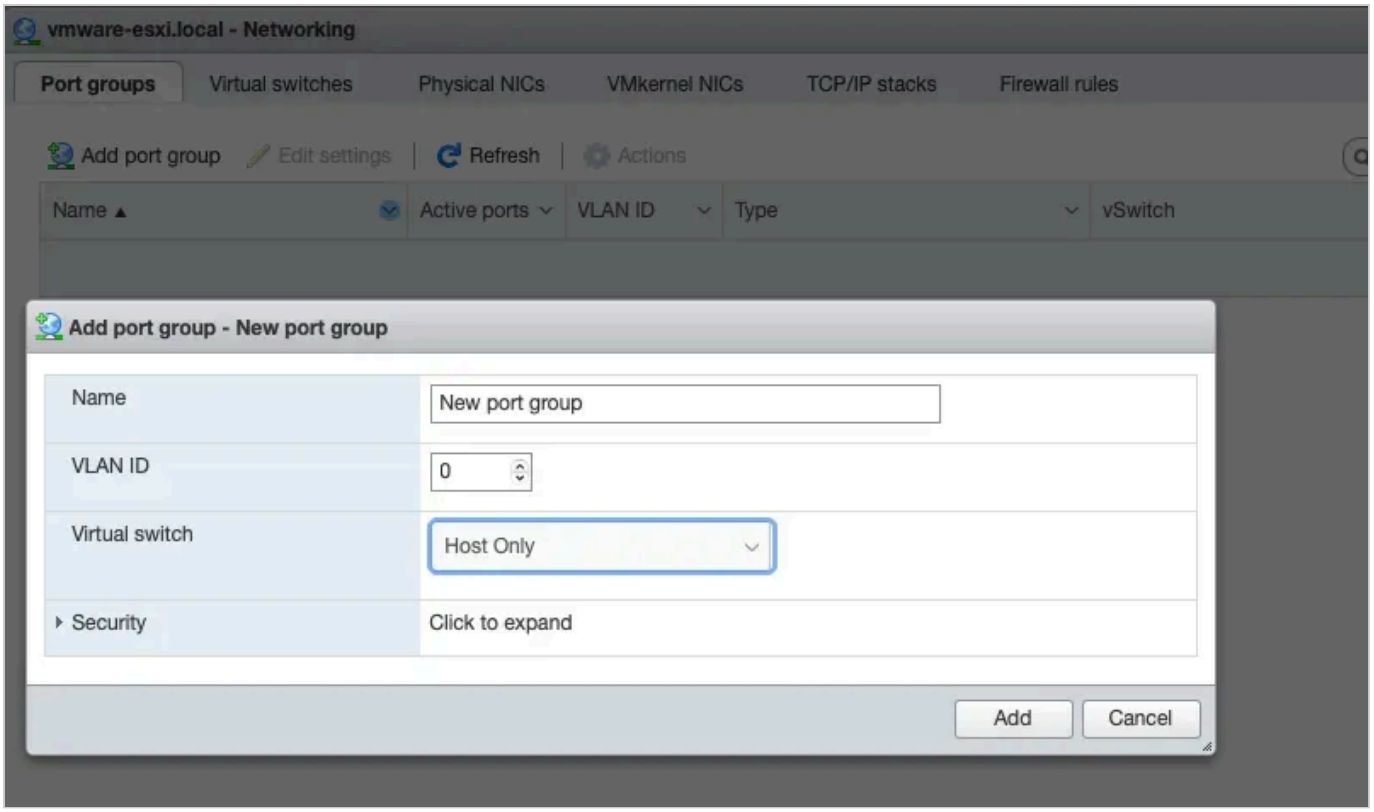
Here is our vSwitch with 0 uplinks.



Success add vswitch

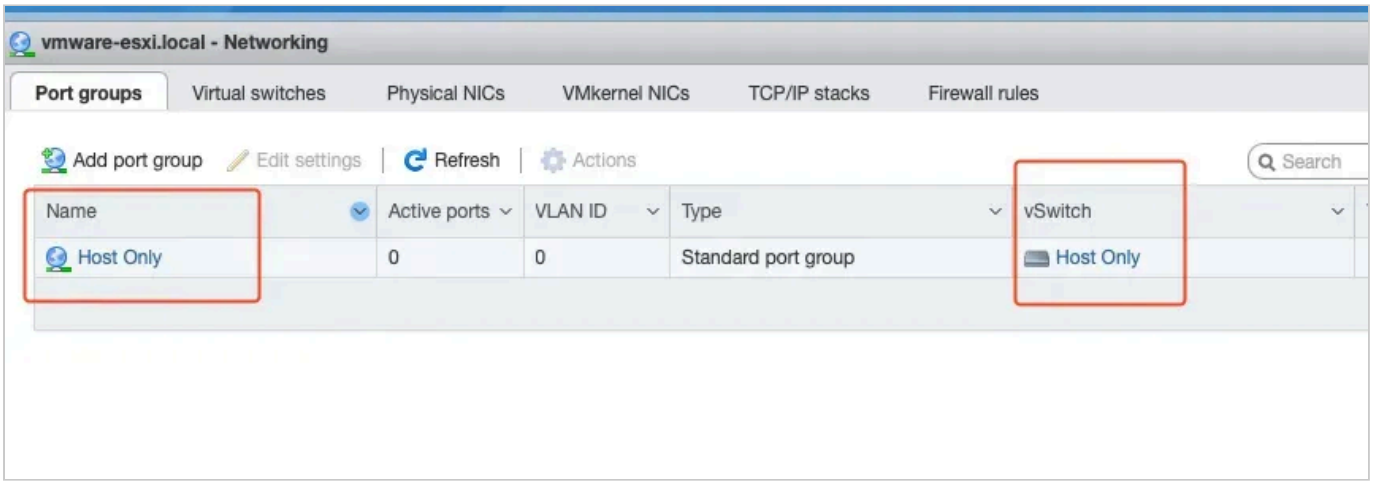
The next step is click Add port group in Port groups tab. Change the name to **Host Only** and Virtual switch to **Host Only** that we created before.

Licensed under CC BY-SA 4.0



Add port group

This is the correct result as expected.

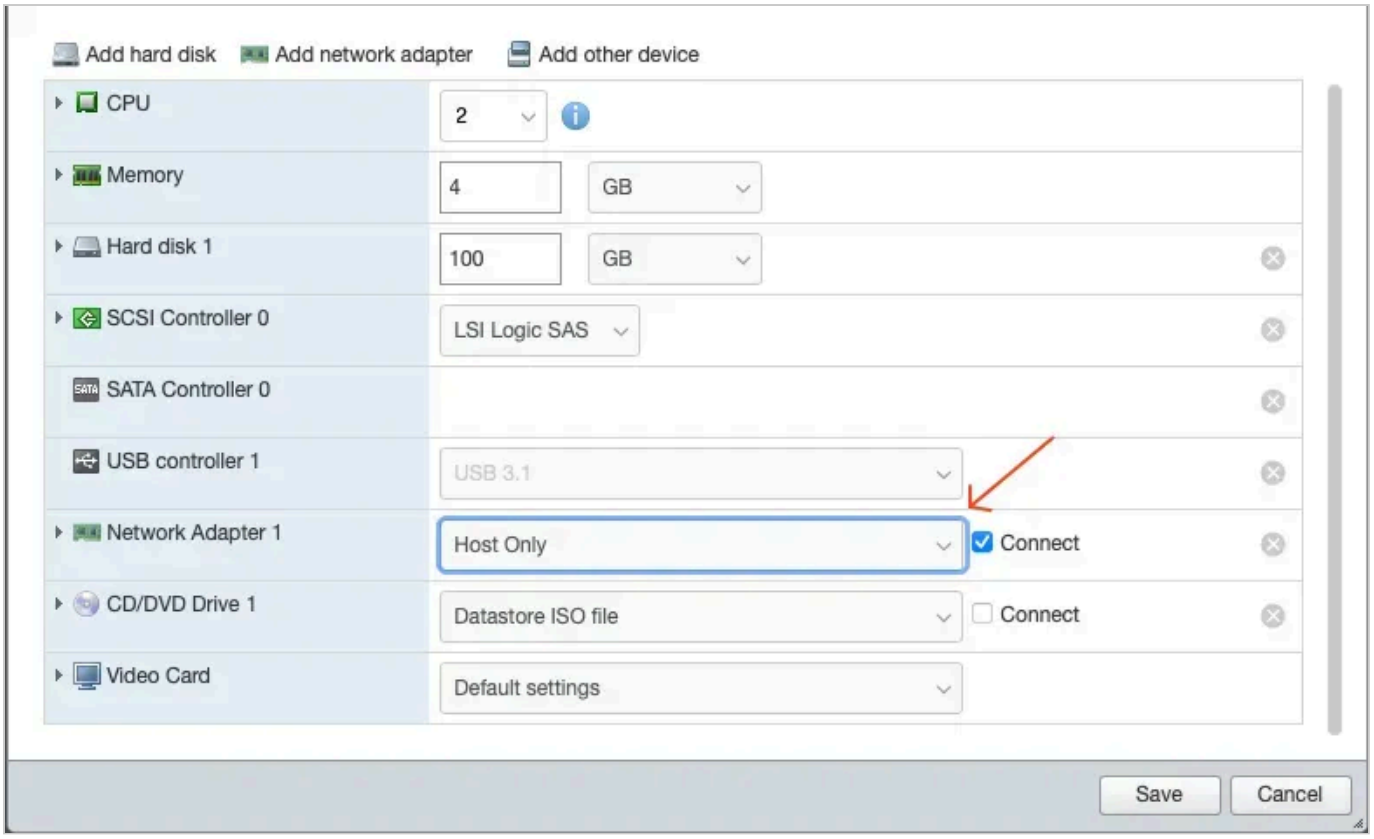


Success add port group

Testing Host Only Network

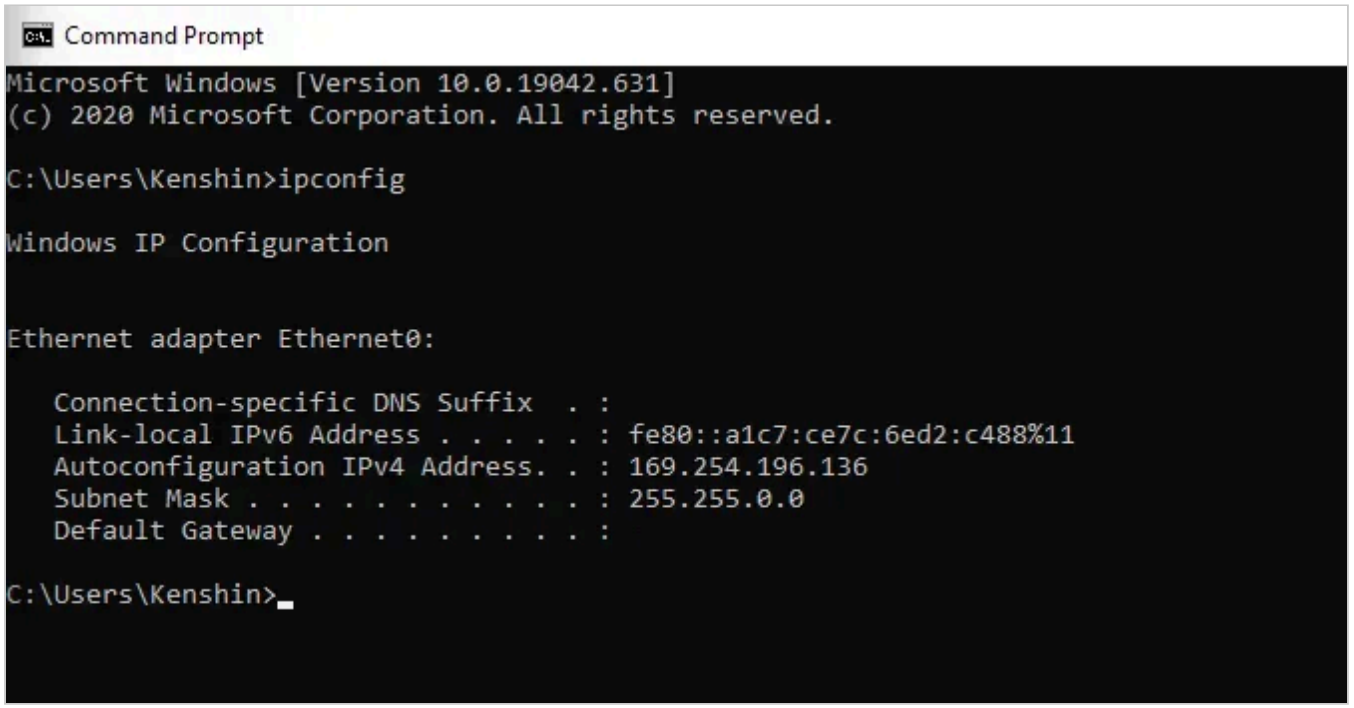
For testing this network, I will use 2 vm: windows and remnux. Make sure that the vm use Host Only as network adapter 1.

Licensed under CC BY-SA 4.0



Change network adapter to Host Only

After booting the vm, we can see that windows and remnux it set to dhcp by default and not getting the ip.



Windows ipconfig

Licensed under CC BY-SA 4.0

```
remnux@remnux:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::20c:29ff:fef6:e4ad  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:f6:e4:ad  txqueuelen 1000  (Ethernet)
        RX packets 117  bytes 12175 (12.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 72  bytes 8362 (8.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 408  bytes 29120 (29.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 408  bytes 29120 (29.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

remnux@remnux:~$
```

Remnux ifconfig

For this testing, we will set the static ip for both vm:

- Windows → 10.10.10.2
- Remnux → 10.10.10.3

Pinging windows to remnux and vice versa

```
Command Prompt

C:\Users\Kenshin>ping 10.10.10.3

Pinging 10.10.10.3 with 32 bytes of data:
Reply from 10.10.10.3: bytes=32 time<1ms TTL=64
Reply from 10.10.10.3: bytes=32 time<1ms TTL=64
Reply from 10.10.10.3: bytes=32 time<1ms TTL=64
Reply from 10.10.10.3: bytes=32 time<1ms TTL=64

Ping statistics for 10.10.10.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Users\Kenshin>
```

Windows ping to remnux

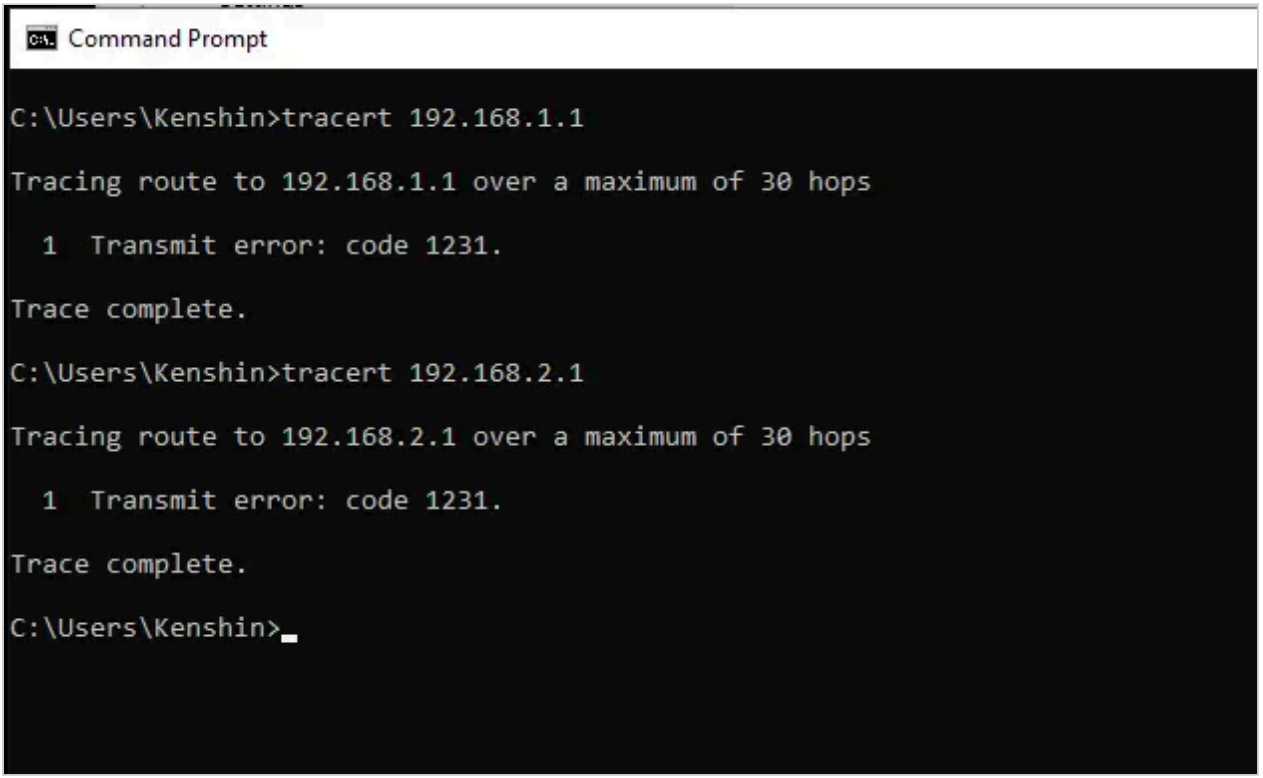
Licensed under CC BY-SA 4.0

```
remnux@remnux:~$ ping 10.10.10.2
PING 10.10.10.2 (10.10.10.2) 56(84) bytes of data.
64 bytes from 10.10.10.2: icmp_seq=1 ttl=128 time=0.322 ms
64 bytes from 10.10.10.2: icmp_seq=2 ttl=128 time=0.193 ms
64 bytes from 10.10.10.2: icmp_seq=3 ttl=128 time=0.237 ms
64 bytes from 10.10.10.2: icmp_seq=4 ttl=128 time=0.204 ms
64 bytes from 10.10.10.2: icmp_seq=5 ttl=128 time=0.209 ms
64 bytes from 10.10.10.2: icmp_seq=6 ttl=128 time=0.186 ms
64 bytes from 10.10.10.2: icmp_seq=7 ttl=128 time=0.202 ms
64 bytes from 10.10.10.2: icmp_seq=8 ttl=128 time=0.207 ms
64 bytes from 10.10.10.2: icmp_seq=9 ttl=128 time=0.207 ms
64 bytes from 10.10.10.2: icmp_seq=10 ttl=128 time=0.257 ms
64 bytes from 10.10.10.2: icmp_seq=11 ttl=128 time=0.186 ms
64 bytes from 10.10.10.2: icmp_seq=12 ttl=128 time=0.191 ms
64 bytes from 10.10.10.2: icmp_seq=13 ttl=128 time=0.206 ms
64 bytes from 10.10.10.2: icmp_seq=14 ttl=128 time=0.181 ms
64 bytes from 10.10.10.2: icmp_seq=15 ttl=128 time=0.204 ms
64 bytes from 10.10.10.2: icmp_seq=16 ttl=128 time=0.221 ms
```

Remnux ping to windows

Ping to other LAN segment

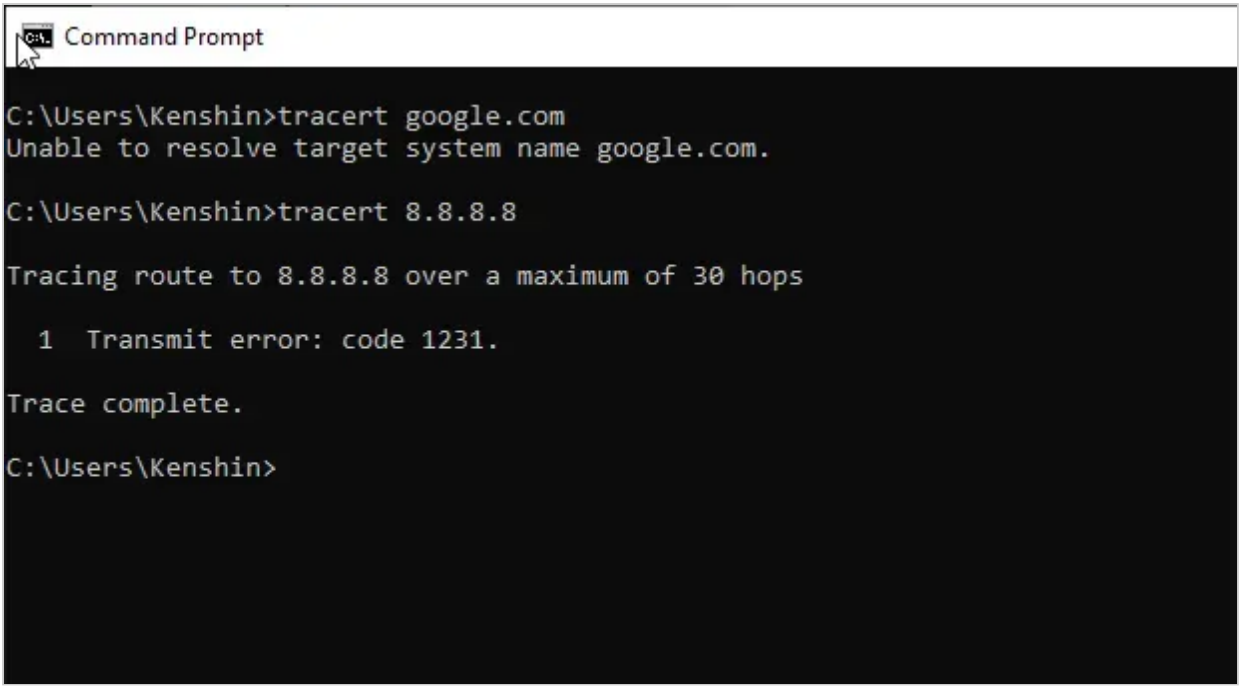
We need to make sure that our real network is not accessible from our malware vm. I try to tracert my LAN segment and it can't access. Good.



Tracert LAN segment

Ping to internet

We need to make sure that vm can't access the internet.



Simulating the network with INetSim

One way to create a simulated network environment for malware analysis is by using INetSim. INetSim is an open-source software that allows you to emulate a variety of network services, including DNS, HTTP, and FTP. By simulating these services, you can create a virtual network environment that closely resembles a real-world network.

On our remnux vm, edit the `/etc/inetsim/inetsim.conf`, uncomment and edit the line

/etc/inetsim/inetsim.conf

```
start_service dns
service_bind_address 0.0.0.0
dns_default_ip       <YOUR_REMNUX_IP>
```

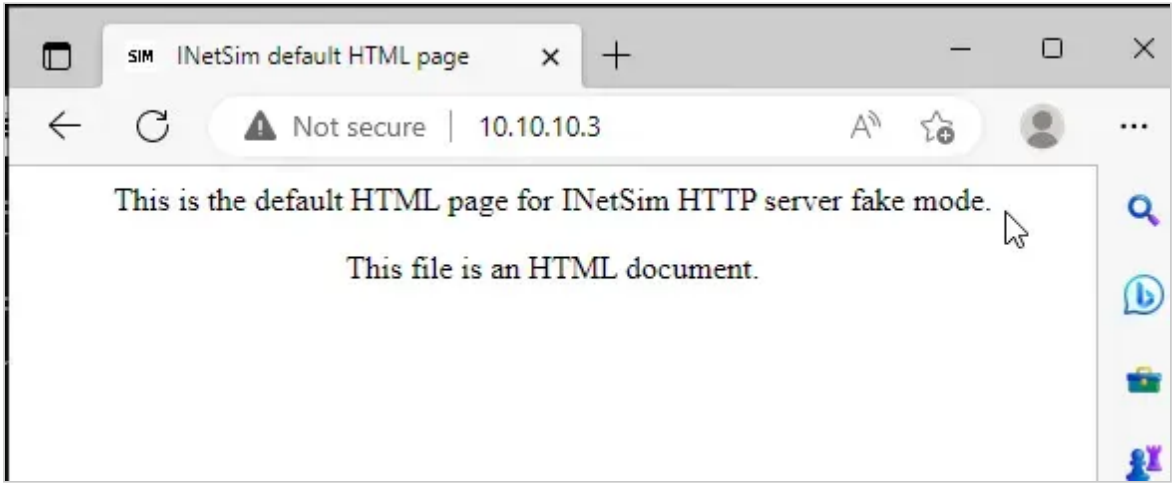
Save and start the inetsim program

Licensed under CC BY-SA 4.0

```
remnux@remnux:~$ inetsim
INetSim 1.3.2 (2020-05-19) by Matthias Eckert & Thomas Hungenberg
Using log directory:      /var/log/inetsim/
Using data directory:     /var/lib/inetsim/
Using report directory:   /var/log/inetsim/report/
Using configuration file: /etc/inetsim/inetsim.conf
Parsing configuration file.
Configuration file parsed successfully.
=== INetSim main process started (PID 1762) ===
Session ID:     1762
Listening on:   10.10.10.3
Real Date/Time: 2023-03-03 09:18:52
Fake Date/Time: 2023-03-03 09:18:52 (Delta: 0 seconds)
 Forking services...
  * dns_53_tcp_udp - started (PID 1766)
  * ftps_990_tcp - started (PID 1774)
  * pop3s_995_tcp - started (PID 1772)
  * smtp_25_tcp - started (PID 1769)
  * http_80_tcp - started (PID 1767)
  * smtps_465_tcp - started (PID 1770)
  * https_443_tcp - started (PID 1768)
  * pop3_110_tcp - started (PID 1771)
  * ftp_21_tcp - started (PID 1773)
 done.
Simulation running.
```

INetSim

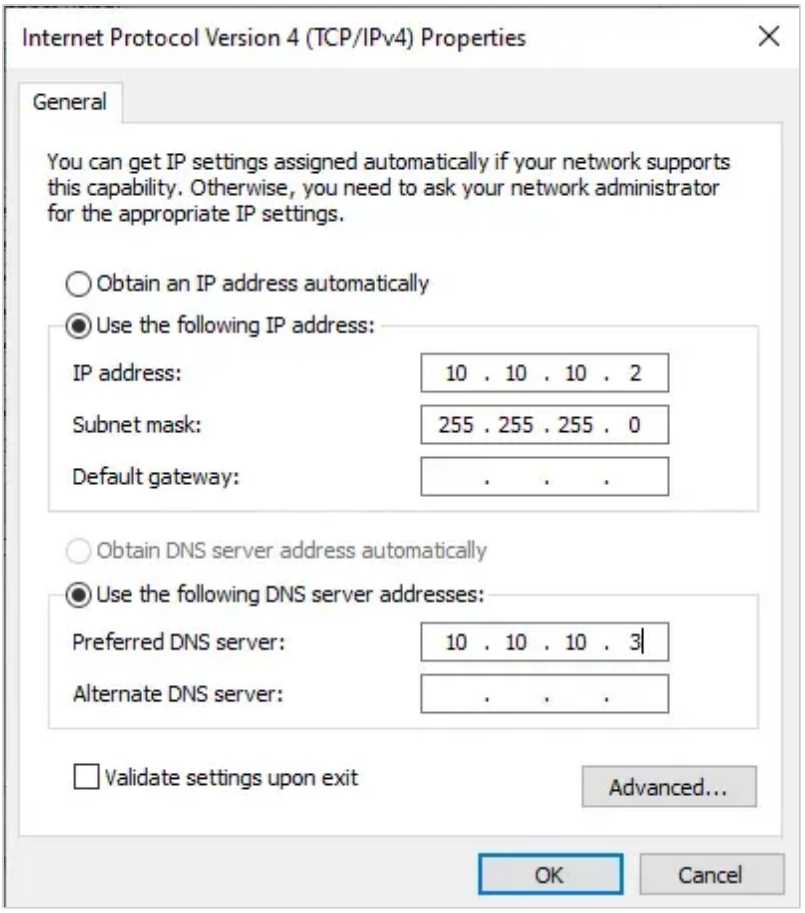
From our windows vm, we try to check the inetsim by accessing **http://REMNUX_IP/** (http://remnux_ip/)



INetSim default page

As you can see, INetSim is success. But, our malware will access the specific domain. To be able to simulate that, we need to set dns address in windows to remnux ip.

Licensed under CC BY-SA 4.0



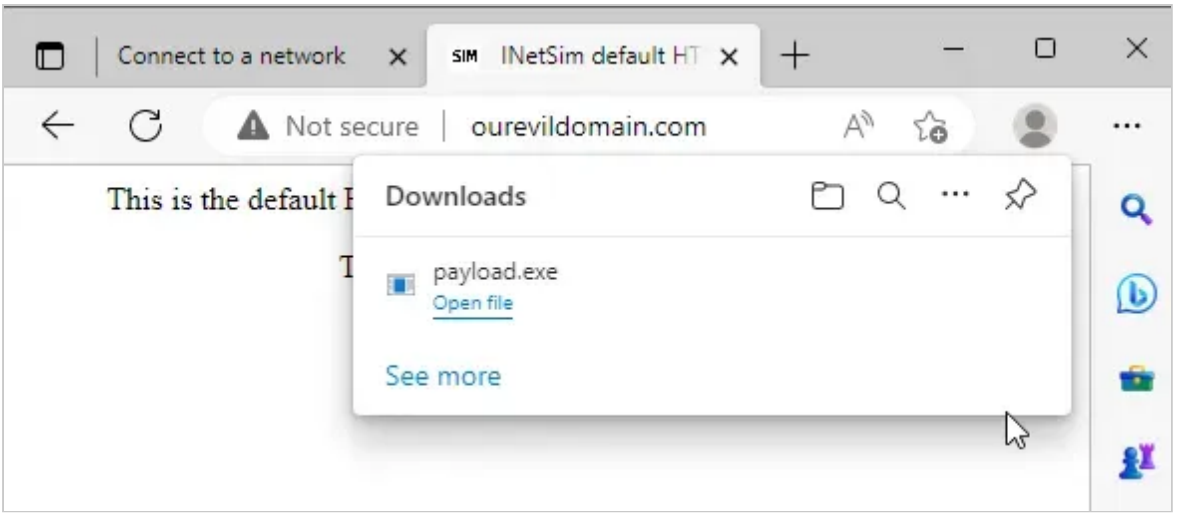
DNS Setup

Check again by accessing ourevildomain.com

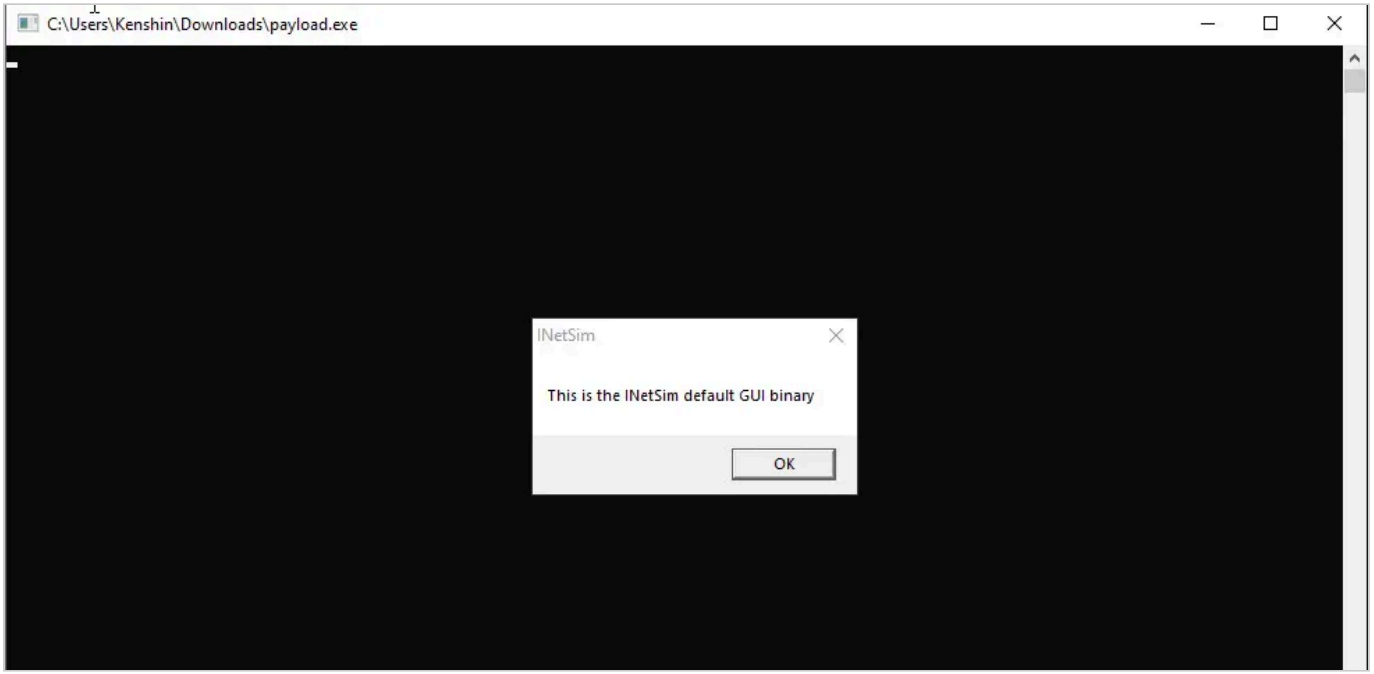


Accessing domain via INetSim

Whatever domain the malware try to access, it will always redirect to our INetSim. If malware try to download.exe, our INetSim will simulate the download process. Let's try to access **http://evil.com/payload.exe** (http://evil.com/payload.exe)



Downloading exe via INetSim



INetSim exe

Summary

- Isolated network environment is necessary for malware analysis.
- The concept of “Host Only” networking can be used in ESXi to create an isolated network environment.
- We can use INetSim to simulate network traffic
- INetSim capable of simulating domain access and file download

Licensed under CC BY-SA 4.0

I hope this information was helpful for understanding the importance of creating an isolated network environment for malware analysis. Always use isolated environment.

Note: In creating this article, I utilized ChatGPT (https://chat.openai.com/), a large language model trained by OpenAI, to assist with writing and editing. ChatGPT was instrumental in ensuring that the content was clear, concise, and easy to understand. Its advanced capabilities allowed me to quickly and efficiently produce high-quality content that is both informative and engaging.

Originally published on Medium (https://medium.com/@petruknisme/malware-analysis-lab-in-esxi-isolated-network-c19bdbdc21ba).

Malware Analysis # Virtualization # Tutorial

Licensed under CC BY-SA 4.0