Lab Exercise2

Lijun Yu

1. Setup two VMs in Azure under the same Network

To meet the prerequisites of this lab, I used Azure to setup instances.

First, I created a resource group:

```
az group create --name CreateVNetQS-rg --location eastus
```

Next, I created a virtual network `myVNet` under the resource group:

```
az network vnet create --name myVNet --resource-group CreateVNetQS-rg --subnet-name
default
```

Then two instances is successfully created:

```
az vm create --resource-group CreateVNetQS-rg --name myVM1 --image UbuntuLTS --
generate-ssh-keys --public-ip-address myPublicIP-myVM1 --no-wait

az vm create --resource-group CreateVNetQS-rg --name myVM2 --image UbuntuLTS --
generate-ssh-keys --public-ip-address myPublicIP-myVM2 --no-wait
```

The result is as followed:

```
❯ az vm list-ip-addresses
[
  {
    "virtualMachine": {
      "name": "myVM1",
      "network": {
        "privateIpAddresses": [
          "10.0.0.4"
        ],
        "publicIpAddresses": [
          {
            "id": "/subscriptions/2abf4044-b29f-4de6-861d-
1aede8b9f933/resourceGroups/CreateVNetQS-
rg/providers/Microsoft.Network/publicIPAddresses/myPublicIP-myVM1",
            "ipAddress": "20.127.40.32",
            "ipAllocationMethod": "Dynamic",
            "name": "myPublicIP-myVM1",
```

```
            "resourceGroup": "CreateVNetQS-rg"
          }
        ]
      },
      "resourceGroup": "CreateVNetQS-rg"
    }
  },
  {
    "virtualMachine": {
      "name": "myVM2",
      "network": {
        "privateIpAddresses": [
          "10.0.0.5"
        ],
        "publicIpAddresses": [
          {
            "id": "/subscriptions/2abf4044-b29f-4de6-861d-
1aede8b9f933/resourceGroups/CreateVNetQS-
rg/providers/Microsoft.Network/publicIPAddresses/myPublicIP-myVM2",
            "ipAddress": "20.127.47.71",
            "ipAllocationMethod": "Dynamic",
            "name": "myPublicIP-myVM2",
            "resourceGroup": "CreateVNetQS-rg"
          }
        ]
      },
      "resourceGroup": "CreateVNetQS-rg"
    }
  }
]
```


As shown above, the instances own the private IP addresses `10.0.0.4` and `10.0.0.5`.

Now let's login to the instances by ssh:

2. Enable Access between instances through Telnet

First, install Telnet:

```
sudo apt-get update && \
     apt install telnetd -y
```

Check the status of the Telnet server:

```
systemctl status inetd
```

```
eileen@myVM2:~$ systemctl status inetd
● inetd.service - Internet superserver
   Loaded: loaded (/lib/systemd/system/inetd.service; enabled; vendor preset:
   Active: active (running) since Sat 2022-02-05 04:48:23 UTC; 1h 2min ago
     Docs: man:inetd(8)
 Main PID: 2702 (inetd)
    Tasks: 1 (limit: 4069)
   CGroup: /system.slice/inetd.service
           └─2702 /usr/sbin/inetd
```

Then verify the accessibility between the two instances through Telnet.

```
eileen@myVM1:~$ telnet 10.0.0.5
Trying 10.0.0.5...
Connected to 10.0.0.5.
Escape character is '^]'.
Ubuntu 18.04.6 LTS
myVM2 login: myvm1
Password:
Last login: Sat Feb  5 05:08:29 UTC 2022 from myvm1.internal.cloudapp.net on
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1067-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sat Feb  5 05:52:28 UTC 2022

  System load:  0.16              Processes:           113
  Usage of /:   5.3% of 28.90GB   Users logged in:     1
  Memory usage: 6%                IP address for eth0: 10.0.0.5
  Swap usage:   0%


33 updates can be applied immediately.
19 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


myvm1@myVM2:~$
```

3. Setup Iptables to block Telnet and Facebook

Add the rule to block outbound request to access telnet server:

```
sudo iptables -A OUTPUT -p tcp --dport telnet -j REJECT
```

The result is shown below as the telnet request is blocked.

```
eileen@myVM1:~$ sudo iptables -A OUTPUT -p tcp --dport telnet -j REJECT
eileen@myVM1:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
REJECT     tcp  --  anywhere             anywhere             tcp dpt:telnet reject-with icmp-port-unreachable
eileen@myVM1:~$ telnet 10.0.0.5
Trying 10.0.0.5...
telnet: Unable to connect to remote host: Connection refused
eileen@myVM1:~$
```

Similarly add another rule to block Facebook.

First check if the website is accessible.

```
eileen@myVM1:~$ http https://www.facebook.com
HTTP/1.1 200 OK
Alt-Svc: h3=":443"; ma=3600, h3-29=":443"; ma=3600
Cache-Control: private, no-cache, no-store, must-revalidate
Connection: keep-alive
Content-Encoding: gzip
Content-Type: text/html; charset="utf-8"
Date: Sat, 05 Feb 2022 06:00:23 GMT
Expires: Sat, 01 Jan 2000 00:00:00 GMT
```

Then check its IP address:

```
eileen@myVM1:~$ dig www.facebook.com

; <<>> DiG 9.11.3-1ubuntu1.16-Ubuntu <<>> www.facebook.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10686
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;www.facebook.com.              IN      A

;; ANSWER SECTION:
www.facebook.com.       283     IN      CNAME   star-mini.c10r.facebook.com.
star-mini.c10r.facebook.com. 2  IN      A       31.13.66.35

;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Sat Feb 05 06:05:45 UTC 2022
;; MSG SIZE  rcvd: 90
```

Add rules by:

```
sudo iptables -A OUTPUT -d 31.13.66.35 -j DROP
```

Now it should not be available.

```
eileen@myVM1:~$ http https://www.facebook.com

http: error: ConnectionError: HTTPSConnectionPool(host='www.facebook.com', port=443): Max retries exceeded with u
(Caused by NewConnectionError('<urllib3.connection.VerifiedHTTPSConnection object at 0x7fd7c344d048>: Failed to e
sh a new connection: [Errno 101] Network is unreachable',)) while doing GET request to URL: https://www.facebook.
eileen@myVM1:~$
```

4. Bypass the Telnet restriction through ssh

To do this, we need another instance, so setup VM3 like VM1 and VM2. Add a user for telnet whose name is `tlenetuser`.

Verifiying the connection, VM1 cannot access VM3 through Telnet, but VM2 can.

```
eileen@myVM1:~/.ssh$ telnet 10.0.0.6
Trying 10.0.0.6...
telnet: Unable to connect to remote host: Connection refused
```

```
eileen@myVM2:~/.ssh$ telnet 10.0.0.6
Trying 10.0.0.6...
Connected to 10.0.0.6.
```

To bypass the Telnet restriction, I setup an ssh tunnel between VM1 and VM2.

```
ssh -L 8000:10.0.0.6:23 10.0.0.5
```

To test if port 8000 is bind:

```
eileen@myVM1:~$ lsof -i :8000
COMMAND   PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
ssh     20445 eileen    4u  IPv6  80808      0t0  TCP ip6-localhost:8000 (LISTEN)
ssh     20445 eileen    5u  IPv4  80809      0t0  TCP localhost:8000 (LISTEN)
```

Try telnet localhost:8000:

```
eileen@myVM1:~$ telnet localhost 8000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Ubuntu 18.04.6 LTS
myVM3 login: telnetuser
Password:
Last login: Sat Feb  5 06:41:56 UTC 2022 from myvm2.internal.cloudapp.net on pts/1
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1067-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sat Feb  5 06:54:54 UTC 2022

  System load:  0.04              Processes:           113
  Usage of /:   5.3% of 28.90GB   Users logged in:     1
  Memory usage: 6%                IP address for eth0: 10.0.0.6
  Swap usage:   0%


33 updates can be applied immediately.
19 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


telnetuser@myVM3:~$
```

Then try to bypass Facefook by:

```
ssh -D 8000 -C 10.0.0.5
```

Since I am using non-GUI Linux, so I used the CLI tool `httpie` to access Facebook, which is also available to introduce socks proxy.

```
http --proxy=http:socks5://localhost:8000 --proxy=https:socks5://localhost:8000
https://www.facebook.com
```

```
eileen@myVM1:~$ http --proxy=http:socks5://localhost:8000 --proxy=https:socks5://localhost:8000 https://www.facebook.com
HTTP/1.1 200 OK
Alt-Svc: h3=":443"; ma=3600, h3-29=":443"; ma=3600
Cache-Control: private, no-cache, no-store, must-revalidate
Connection: keep-alive
Content-Encoding: gzip
Content-Type: text/html; charset="utf-8"
Date: Sun, 06 Feb 2022 02:02:30 GMT
Expires: Sat, 01 Jan 2000 00:00:00 GMT
Pragma: no-cache
Priority: u=3,i
```

5. Bypass an internal web server by a reverse ssh tunnel

Background: I have a simple http server running on VM1 port 8080. Any inbound request at port 22 and 8080 is rejected. I also have a VM2 with a ssh account to get access to VM1(which is not available now due to the firewall). My host is able to access VM2 through ssh. The goal is to use my host to access the internal web service of VM1.

First, setup a reverse ssh tunnel between VM1 and VM2.

```
ssh -f -N -T -R22222:localhost:22 10.0.0.5
```

VM1 sent the request to VM2 for ssh connection on VM2's port 22222. Then VM2 can access VM1 through its own port 22222. That way bypass the firewall.

Then bind VM2's port 9000 with 'localhost:8080' at port 22222 so as to access VM1's internal web service.

```
ssh -N -L 9000:127.0.0.1:8080 -p 22222 localhost
```

Finally bind my host's port 9001 with the ssh tunnel to VM2.

```
ssh -D 9001 20.127.47.71
```

So now I can get access to port 8080 of VM1 through the following command:

```
http --proxy=http:socks5://localhost:9001 http://127.0.0.1:9000/key
```

```
❯ http --proxy=http:socks5://localhost:9001 http://127.0.0.1:9000/key
HTTP/1.0 200 OK
Content-Length: 14
Content-type: application/octet-stream
Date: Sun, 06 Feb 2022 03:08:12 GMT
Last-Modified: Sat, 05 Feb 2022 07:01:09 GMT
Server: SimpleHTTP/0.6 Python/3.6.9

So say we are
```

6. Netfilter through LKM

Below is the implementation of a simple Linux Lodable Kernal Module to block inbound requests to port 8080.

```
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/ip.h>
#include <linux/tcp.h>
#include <linux/udp.h>

static struct nf_hook_ops *nfho = NULL;

static unsigned char *ip_address = "\x7F\x00\x00\x01"; // Localhost: 127.0.0.1
```

```c
static unsigned int hfunc(void *priv, struct sk_buff *skb, const struct nf_hook_state
*state)
{
  if (!skb)
    return NF_ACCEPT;

  struct iphdr *iph;
  struct tcphdr *tcph;

  iph = ip_hdr(skb);

  // Accept request from localhost
  if(iph->saddr == *(unsigned int*)ip_address){
    return NF_ACCEPT;
  }

  // Block request over 8080
  if (iph->protocol == IPPROTO_TCP) {
    tcph = tcp_hdr(skb);
    if (ntohs(tcph->dest) == 8080) {
      return NF_DROP;
    }
  }

  return NF_ACCEPT;
}

static int __init LKM_init(void)
{
  nfho = (struct nf_hook_ops*)kcalloc(1, sizeof(struct nf_hook_ops), GFP_KERNEL);

  /* Initialize netfilter hook */
  nfho->hook  = (nf_hookfn*)hfunc;    /* hook function */
  nfho->hooknum   = NF_INET_PRE_ROUTING;    /* received packets */
  nfho->pf  = PF_INET;      /* IPv4 */
  nfho->priority  = NF_IP_PRI_FIRST;    /* max hook priority */

  int ret = nf_register_net_hook(&init_net, nfho);
  return ret;
}

static void __exit LKM_exit(void)
{
  nf_unregister_net_hook(&init_net, nfho);
  kfree(nfho);
}

module_init(LKM_init);
```

```
module_exit(LKM_exit);
```

Now in VM2, I cannot access VM1(10.0.0.4)'s port 8080. But still since I'm using the ssh to bypass the firewall, I can access it art my host.

To test the result, I installed a Tshark, which is the headless Wireshark.

If send normal request, the result would be like:

```
eileen@myVM1:~$ sudo tshark -Y "tcp.dstport == 8080"
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
   39 5.250645016     10.0.0.5 → 10.0.0.4     TCP 74 60516 → 8080 [SYN]
Seq=0 Win=64240 Len=0 MSS=1418 SACK_PERM=1 TSval=3290487296 TSecr=0 WS=1
28
   44 6.253950270     10.0.0.5 → 10.0.0.4     TCP 74 [TCP Retransmission
] 60516 → 8080 [SYN] Seq=0 Win=64240 Len=0 MSS=1418 SACK_PERM=1 TSval=32
90488300 TSecr=0 WS=128
   83 8.269929842     10.0.0.5 → 10.0.0.4     TCP 74 [TCP Retransmission
] 60516 → 8080 [SYN] Seq=0 Win=64240 Len=0 MSS=1418 SACK_PERM=1 TSval=32
90490316 TSecr=0 WS=128
   93 12.397894082     10.0.0.5 → 10.0.0.4     TCP 74 [TCP Retransmissi$
```

But if try to bypass, the result is:

```
  111 10.195935036     10.0.0.4 → 10.0.0.5     SSH 150 Client: Encrypted
 packet (len=84)
  116 10.197743968     10.0.0.4 → 10.0.0.5     SSH 390 Client: Encrypted
 packet (len=324)
  119 10.198502582     10.0.0.4 → 10.0.0.5     SSH 142 Client: Encrypted
 packet (len=76)
```