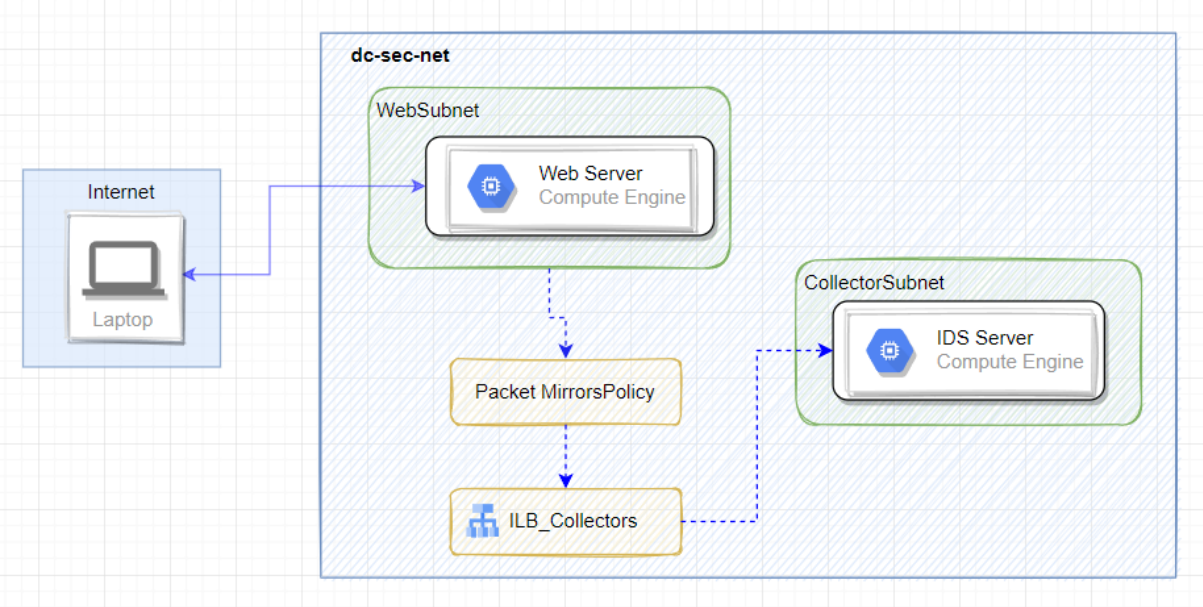
Packet Mirrors && IDS

Created	@March 2, 2022 11:35 AM
Tags	IDS Packet Mirrors
Modify Date	@March 2, 2022
Description	通过packet Mirrors实现流量镜像
Link	

一、概述：



二、具体部署过程：

- 创建VPC 网络
  - 创建Web Server Subnet
  - 创建 Collector Subnet
- 配置网络防火墙与Cloud NAT
- 创建Web server
- 创建IDS server
- 创建ILB Collector
- 测试验证
  - 测试IDS server 正常接收packet mirrors流量
  - 测试Web server egress Traffic
  - 测试Web server ingress Traffic

2.1 创建一个VPC

```
$ gcloud compute networks create dm-sec-net --subnet-mode=custom
Created [https://www.googleapis.com/compute/v1/projects/yunion-test-286209/global/networks/dm-sec-net].
NAME: dm-sec-net
SUBNET_MODE: CUSTOM
BGP_ROUTING_MODE: REGIONAL
IPV4_RANGE:
GATEWAY_IPV4:

Instances on this network will not be reachable until firewall rules
are created. As an example, you can allow all internal traffic between
instances as well as SSH, RDP, and ICMP by running:

$ gcloud compute firewall-rules create <FIREWALL_NAME> --network dm-sec-net --allow tcp,udp,icmp --source-ranges <IP_RANGE>
$ gcloud compute firewall-rules create <FIREWALL_NAME> --network dm-sec-net --allow tcp:22,tcp:3389,icmp
```

2.2 在us-central1 为packet Mirrors 添加一个subnet

```
$ gcloud compute networks subnets create dm-sec-net-uscentral1 \
> --range=172.21.0.0/24 \
> --network=dm-sec-net \
> --region=us-central1

Created [https://www.googleapis.com/compute/v1/projects/yunion-test-286209/regions/us-central1/subnetworks/dm-sec-net-uscentral1].
NAME: dm-sec-net-uscentral1
REGION: us-central1
NETWORK: dm-sec-net
RANGE: 172.21.0.0/24
STACK_TYPE: IPV4_ONLY
IPV6_ACCESS_TYPE:
IPV6_CIDR_RANGE:
EXTERNAL_IPV6_CIDR_RANGE:
```

2.3 在 us-central1 为采集器创建一个subnet ：

```
$ gcloud compute networks subnets create dm-sec-net-uscentral1-ids --range=172.21.1.0/24 --network=dm-sec-net --region=us-central1

Created [https://www.googleapis.com/compute/v1/projects/yunion-test-286209/regions/us-central1/subnetworks/dm-sec-net-uscentral1-ids].
NAME: dm-sec-net-uscentral1-ids
REGION: us-central1
NETWORK: dm-sec-net
RANGE: 172.21.1.0/24
STACK_TYPE: IPV4_ONLY
IPV6_ACCESS_TYPE:
IPV6_CIDR_RANGE:
EXTERNAL_IPV6_CIDR_RANGE:
```

2.4 创建防火墙规则和NAT：

- 规则 1：允许来自所有来源的所有虚拟机使用标准 http 端口 (TCP 80) 和 ICMP 协议
- 规则 2： 允许 IDS 接收来自所有源的所有流量。注意在后面的部分中为 IDS VM 无外网 IP。
- 规则 3： 允许“Google Cloud IAP 代理”IP 范围 TCP 端口 22 到所有虚拟机，使能够通过 Cloud Console SSH 进入虚拟机

```
# Rule 1:
gcloud compute firewall-rules create fw-dm-secnet-allow-any-web \
--direction=INGRESS \
--priority=1000 \
--network=dm-sec-net \
--action=ALLOW \
--rules=tcp:80,icmp \
--source-ranges=0.0.0.0/0
```

```
gcloud compute firewall-rules create fw-dm-secnet-ids-any-any \
--direction=INGRESS \
--priority=1000 \
--network=dm-sec-net \
--action=ALLOW \
--rules=all \
--source-ranges=0.0.0.0/0 \
--target-tags=ids
```

```
gcloud compute firewall-rules create fw-dm-secnet-iapproxy \
--direction=INGRESS \
--priority=1000 \
--network=dm-sec-net \
--action=ALLOW \
--rules=tcp:22,icmp \
--source-ranges=35.235.240.0/20
```

2.5 创建一个CloudRouter：

```
gcloud compute routers create router-secnet-nat-central1 \
--region=us-central1 \
--network=dm-sec-net
```

2.6 创建一个Cloud NAT

```
gcloud compute routers nats create nat-gw-dm-secnet-central1 \
--router=router-secnet-nat-central1 \
--router-region=us-central1 \
--auto-allocate-nat-external-ips \
--nat-all-subnet-ip-ranges
```

2.7 创建 Web Server 实例模板：

```
gcloud compute instance-templates create template-dm-secnet-web-us-central1 \
--region=us-central1 \
--network=dm-sec-net \
--subnet=dm-sec-net-uscentral1 \
--machine-type=g1-small \
--image=ubuntu-1604-xenial-v20200807 \
--image-project=ubuntu-os-cloud \
--tags=webserver \
--metadata=startup-script='#! /bin/bash
  apt-get update
  apt-get install apache2 -y
  vm_hostname="$(curl -H "Metadata-Flavor:Google" \
  http://169.254.169.254/computeMetadata/v1/instance/name)"
  echo "Page served from: $vm_hostname" | \
  tee /var/www/html/index.html
  systemctl restart apache2'
```

2.8 基于实例模板创建Web Server 实例组：

```
gcloud compute instance-groups managed create mig-dm-secnet-web-uscentral1 \
    --template=template-dm-secnet-web-us-central1 \
    --size=2 \
    --zone=us-central1-a
```

2.9 创建IDS Server 实例模板：

```
gcloud compute instance-templates create template-dm-secnet-ids-us-central1 \
--region=us-central1 \
--network=dm-sec-net \
--no-address \
--subnet=dm-sec-net-uscentral1-ids \
--image=ubuntu-1604-xenial-v20200807 \
--image-project=ubuntu-os-cloud \
--tags=ids,webserver \
--metadata=startup-script='#! /bin/bash
  apt-get update
  apt-get install apache2 -y
  vm_hostname="$(curl -H "Metadata-Flavor:Google" \
  http://169.254.169.254/computeMetadata/v1/instance/name)"
  echo "Page served from: $vm_hostname" | \
  tee /var/www/html/index.html
  systemctl restart apache2'
```

2.10 创建IDS Server 实例组：

```
gcloud compute instance-groups managed create mig-dm-secnet-ids-uscentral1 \
    --template=template-dm-secnet-ids-us-central1 \
    --size=1 \
    --zone=us-central1-a
```

创建内部负载均衡（Internal LoadBalancer）

Packet Mirrors 通过ILB 转发所有镜像流量到后端收集器，本Demo 实例组包含一个VM，可以根据实际配置为自动扩缩实例组：

- 创建后端服务（BackendService）health Check

```
gcloud compute health-checks create tcp hc-tcp-80 --port 80
```

- 创建ILB Backend Service：

```
gcloud compute backend-services create be-dm-secnet-suricata-us-central1 \
--load-balancing-scheme=INTERNAL \
--health-checks=hc-tcp-80 \
--network=dm-sec-net \
--protocol=TCP \
--region=us-central1
```

- 添加创建好的IDS 管理实例组到后端服务：

```
gcloud compute backend-services add-backend be-dm-secnet-suricata-us-central1 \
--instance-group=mig-dm-secnet-ids-uscentral1 \
--instance-group-zone=us-central1-a \
--region=us-central1
```

- 创建一个前端转发规则作为后端收集器收集点：

```
gcloud compute forwarding-rules create ilb-dm-secnet-suricata-ilb-us-central1 \
 --load-balancing-scheme=INTERNAL \
 --backend-service be-dm-secnet-suricata-us-central1 \
 --is-mirroring-collector \
 --network=dm-sec-net \
 --region=us-central1 \
 --subnet=dm-sec-net-uscentral1-ids \
 --ip-protocol=TCP \
 --ports=all

注意：配置  --is-mirroring-collector  ## flag
```

在IDS collector VM上，安装开源 IDS - Suricata：

- IAP 方式连接到VM：

```
$ gcloud compute ssh  mig-dm-secnet-ids-uscentral1-4rsz --zone us-central1-a --tunnel-through-iap
```

- 更新IDS VM：

```
sudo apt-get update -y
```

- 安装 Suricata 依赖环境

```
sudo apt-get install libpcre3-dbg libpcre3-dev autoconf automake libtool libpcap-dev libnet1-dev libyaml-dev zlib1g-dev libcap-ng-dev
```

```
sudo apt-get install libnspr4-dev -y

sudo apt-get install libnss3-dev -y
```

```
sudo apt-get install liblz4-dev -y

sudo apt install rustc cargo -y
```

安装Suricata

```
sudo add-apt-repository ppa:oisf/suricata-stable -y
```

```
# 更新source
sudo apt-get update -y
```

```
sudo apt-get install suricata -y
```

验证 Suricata 安装：

```
## 输出以下信息，证明Suricata 安装完成
$ suricata -V
This is Suricata version 6.0.3 RELEASE
```

停止Suricata 服务，并备份Suricata 默认配置文件：

```
$ sudo systemctl stop suricata

$ sudo cp /etc/suricata/suricata.{yaml,bakup}
```

修改suricata 配置文件和规则文件：

```
~# sudo mkdir /etc/suricata/poc-rules
~# sudo cp suricata_Conf/my.rules /etc/suricata/poc-rules/my.rules
~# sudo cp suricata_Conf/suricata.yaml /etc/suricata/poc-rules/suricata.yaml
```

启动Suricata 服务：

```
$ sudo systemctl start suricata
~# systemctl status suricata
● suricata.service - LSB: Next Generation IDS/IPS
   Loaded: loaded (/etc/init.d/suricata; bad; vendor preset: enabled)
   Active: active (exited) since Wed 2022-03-02 04:35:53 UTC; 13s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 11481 ExecStop=/etc/init.d/suricata stop (code=exited, status=1/FAILURE)
  Process: 11686 ExecStart=/etc/init.d/suricata start (code=exited, status=0/SUCCESS)

Mar 02 04:35:53 mig-dm-secnet-ids-uscentral1-4rsz systemd[1]: Starting LSB: Next Generation IDS/IPS...
```

查看 Rule

```
$ cat /etc/suricata/poc-rules/my.rules
####RULES#####
#UDP ALERTS
alert udp $HOME_NET any -> 8.8.8.8 53 (msg:"BAD UDP DNS REQUEST"; sid:99996; rev:1;)

#HTTP ALERTS
alert http any any -> $HOME_NET 80 (msg:"BAD HTTP PHP REQUEST"; http.uri; content:"index.php"; sid:99997; rev:1;)

#ICMP ALERTS
alert icmp any any -> $HOME_NET any (msg:"BAD ICMP"; sid:99998; rev:1;)

#TCP ALERTS
alert tcp $HOME_NET any -> any 6667 (msg:"BAD TCP 6667 REQUEST"; sid:99999; rev:1;)
```

更多Rule 规则参考：

```
# cat /etc/suricata/rules/
app-layer-events.rules  dhcp-events.rules       dns-events.rules        http-events.rules       kerberos-events.rules   nfs-events.rul
decoder-events.rules    dnp3-events.rules       files.rules             ipsec-events.rules      modbus-events.rules     ntp-events.rul
```

配置Packet Mirrors 策略：

```
gcloud compute packet-mirrorings create mirror-dm-secnet-web \
--collector-ilb=ilb-dm-secnet-suricata-ilb-us-central1 \
--network=dm-sec-net \
--mirrored-subnets=dm-sec-net-uscentral1 \
--region=us-central1
```

三、测试验证Packet Mirrors 镜像流量

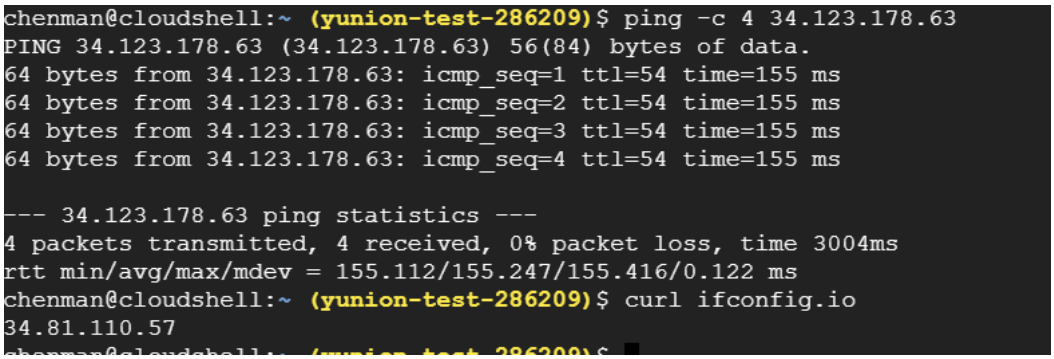
测试验证packet Mirrors ：

- 登录IDS VM/suricata :

```
$ gcloud compute ssh  mig-dm-secnet-ids-uscentral1-4rsz --zone us-central1-a --tunnel-through-iap
```

在IDS 上使用 tcpdump 进行抓包；然后在cloudShell 中通过ping 第一个Web Server Public IP，验证ILB正常将流量镜像给了IDS Server，抓包详细信息如下：

```
$ sudo tcpdump -i ens4 -nn -n "(icmp or port 80) and net 172.21.0.0/24"
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens4, link-type EN10MB (Ethernet), capture size 262144 bytes
04:47:44.038747 IP 172.172.23.208.11827 > 172.21.0.2.80: Flags [S], seq 601843981, win 57493, length 0
04:47:44.038998 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], len
04:47:45.050711 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], len
04:47:47.066686 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], len
04:47:51.194659 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], len
04:47:59.387049 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], len
04:48:15.515172 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], len
04:48:30.725974 IP 34.81.110.57 > 172.21.0.3: ICMP echo request, id 64553, seq 1, length 64
04:48:30.726218 IP 172.21.0.3 > 34.81.110.57: ICMP echo reply, id 64553, seq 1, length 64
04:48:31.726764 IP 34.81.110.57 > 172.21.0.3: ICMP echo request, id 64553, seq 2, length 64
04:48:31.727182 IP 172.21.0.3 > 34.81.110.57: ICMP echo reply, id 64553, seq 2, length 64
04:48:32.728203 IP 34.81.110.57 > 172.21.0.3: ICMP echo request, id 64553, seq 3, length 64
04:48:32.728625 IP 172.21.0.3 > 34.81.110.57: ICMP echo reply, id 64553, seq 3, length 64
04:48:33.729400 IP 34.81.110.57 > 172.21.0.3: ICMP echo request, id 64553, seq 4, length 64
04:48:33.729787 IP 172.21.0.3 > 34.81.110.57: ICMP echo reply, id 64553, seq 4, length 64
```



```
chenman@mig-dm-secnet-ids-uscentral1-4rsz:~$ sudo tcpdump -i ens4 -nn -n "(icmp or port 80) and net 172.21.0.0/24"
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens4, link-type EN10MB (Ethernet), capture size 262144 bytes
04:47:44.038747 IP 172.172.23.208.11827 > 172.21.0.2.80: Flags [S], seq 601843981, win 57493, length 0
04:47:44.038998 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], length 0
04:47:45.050711 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], length 0
04:47:47.066686 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], length 0
04:47:51.194659 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], length 0
04:47:59.387049 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], length 0
04:48:15.515172 IP 172.21.0.2.80 > 172.172.23.208.11827: Flags [S.], seq 4270855113, ack 601843982, win 65320, options [mss 1420], length 0
04:48:30.725974 IP 34.81.110.57 > 172.21.0.3: ICMP echo request, id 64553, seq 1, length 64
04:48:30.726218 IP 172.21.0.3 > 34.81.110.57: ICMP echo reply, id 64553, seq 1, length 64
04:48:31.726764 IP 34.81.110.57 > 172.21.0.3: ICMP echo request, id 64553, seq 2, length 64
04:48:31.727182 IP 172.21.0.3 > 34.81.110.57: ICMP echo reply, id 64553, seq 2, length 64
04:48:32.728203 IP 34.81.110.57 > 172.21.0.3: ICMP echo request, id 64553, seq 3, length 64
04:48:32.728625 IP 172.21.0.3 > 34.81.110.57: ICMP echo reply, id 64553, seq 3, length 64
04:48:33.729400 IP 34.81.110.57 > 172.21.0.3: ICMP echo request, id 64553, seq 4, length 64
04:48:33.729787 IP 172.21.0.3 > 34.81.110.57: ICMP echo reply, id 64553, seq 4, length 64
04:54:03.950939 IP 34.81.110.57 > 172.21.0.2: ICMP echo request, id 50033, seq 1, length 64
04:54:03.951248 IP 172.21.0.2 > 34.81.110.57: ICMP echo reply, id 50033, seq 1, length 64
04:54:04.951974 IP 34.81.110.57 > 172.21.0.2: ICMP echo request, id 50033, seq 2, length 64
04:54:04.952442 IP 172.21.0.2 > 34.81.110.57: ICMP echo reply, id 50033, seq 2, length 64
04:54:05.953306 IP 34.81.110.57 > 172.21.0.2: ICMP echo request, id 50033, seq 3, length 64
04:54:05.953846 IP 172.21.0.2 > 34.81.110.57: ICMP echo reply, id 50033, seq 3, length 64
04:54:06.954580 IP 34.81.110.57 > 172.21.0.2: ICMP echo request, id 50033, seq 4, length 64
04:54:06.955158 IP 172.21.0.2 > 34.81.110.57: ICMP echo reply, id 50033, seq 4, length 64
```

测试http Request：

```
chenman@cloudshell:~ (yunion-test-286209)$ curl -I http://34.123.178.63
HTTP/1.1 200 OK
Date: Wed, 02 Mar 2022 04:55:16 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Wed, 02 Mar 2022 04:05:35 GMT
ETag: "34-5d93464c2061a"
Accept-Ranges: bytes
Content-Length: 52
Content-Type: text/html

chenman@cloudshell:~ (yunion-test-286209)$ curl ifconfig.io
34.81.110.57
```

IDS VM：

```
04:55:16.583447 IP 34.81.110.57.38152 > 172.21.0.3.80: Flags [S], seq 2990226645, win 64240, options [mss 1460,sackOK,TS val 833554243 ecr 0,nop,wscale 7], length 0
04:55:16.583726 IP 172.21.0.3.80 > 34.81.110.57.38152: Flags [S.], seq 404038256, ack 2990226646, win 64768, options [mss 1420,sackOK,TS val 2230345540 ecr 833554243,nop,wscale 7], length 0
04:55:16.737340 IP 34.81.110.57.38152 > 172.21.0.3.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 833554399 ecr 2230345540], length 0
04:55:16.737362 IP 34.81.110.57.38152 > 172.21.0.3.80: Flags [P.], seq 1:79, ack 1, win 502, options [nop,nop,TS val 833554399 ecr 2230345540], length 78: HTTP: HEAD / HTTP/1.1
04:55:16.737466 IP 172.21.0.3.80 > 34.81.110.57.38152: Flags [.], ack 79, win 506, options [nop,nop,TS val 2230345695 ecr 833554399], length 0
04:55:16.738032 IP 172.21.0.3.80 > 34.81.110.57.38152: Flags [P.], seq 1:228, ack 79, win 506, options [nop,nop,TS val 2230345695 ecr 833554399], length 227: HTTP: HTTP/1.1 200 OK
04:55:16.891400 IP 34.81.110.57.38152 > 172.21.0.3.80: Flags [.], ack 228, win 501, options [nop,nop,TS val 833554553 ecr 2230345695], length 0
04:55:16.891641 IP 34.81.110.57.38152 > 172.21.0.3.80: Flags [F.], seq 79, ack 228, win 501, options [nop,nop,TS val 833554553 ecr 2230345695], length 0
04:55:16.891885 IP 172.21.0.3.80 > 34.81.110.57.38152: Flags [F.], seq 228, ack 80, win 506, options [nop,nop,TS val 2230345849 ecr 833554553], length 0
04:55:17.045441 IP 34.81.110.57.38152 > 172.21.0.3.80: Flags [.], ack 229, win 501, options [nop,nop,TS val 833554707 ecr 2230345849], length 0
```

通过本地浏览器访问：

访问Web-1：

```
04:56:39.681042 IP 167.179.68.23.53500 > 172.21.0.3.80: Flags [S], seq 3408460250, win 64240, options [mss 1400,nop,wscale 8,nop,nop,sackOK], length 0
04:56:39.681291 IP 172.21.0.3.80 > 167.179.68.23.53500: Flags [S.], seq 4078977089, ack 3408460251, win 65320, options [mss 1420,nop,nop,sackOK,nop,wscale 7], length 0
04:56:39.681491 IP 167.179.68.23.53501 > 172.21.0.3.80: Flags [S], seq 2482258213, win 64240, options [mss 1400,nop,wscale 8,nop,nop,sackOK], length 0
04:56:39.681814 IP 172.21.0.3.80 > 167.179.68.23.53501: Flags [S.], seq 175704720, ack 2482258214, win 65320, options [mss 1420,nop,nop,sackOK,nop,wscale 7], length 0
04:56:39.913187 IP 167.179.68.23.53500 > 172.21.0.3.80: Flags [.], ack 1, win 514, length 0
04:56:39.913461 IP 167.179.68.23.53500 > 172.21.0.3.80: Flags [P.], seq 1:433, ack 1, win 514, length 432: HTTP: GET / HTTP/1.1
04:56:39.913549 IP 172.21.0.3.80 > 167.179.68.23.53500: Flags [.], ack 433, win 507, length 0
04:56:39.913992 IP 172.21.0.3.80 > 167.179.68.23.53500: Flags [P.], seq 1:336, ack 433, win 507, length 335: HTTP: HTTP/1.1 200 OK
04:56:39.914333 IP 167.179.68.23.53501 > 172.21.0.3.80: Flags [.], ack 1, win 514, length 0
04:56:40.193335 IP 167.179.68.23.53500 > 172.21.0.3.80: Flags [.], ack 336, win 512, length 0
04:56:40.228111 IP 167.179.68.23.53500 > 172.21.0.3.80: Flags [P.], seq 433:800, ack 336, win 512, length 367: HTTP: GET /null HTTP/1.1
04:56:40.228406 IP 172.21.0.3.80 > 167.179.68.23.53500: Flags [P.], seq 336:827, ack 800, win 505, length 491: HTTP: HTTP/1.1 404 Not Found
04:56:40.469176 IP 167.179.68.23.53500 > 172.21.0.3.80: Flags [P.], seq 800:1174, ack 827, win 510, length 374: HTTP: GET /favicon.ico HTTP/1.1
04:56:40.469455 IP 172.21.0.3.80 > 167.179.68.23.53500: Flags [P.], seq 827:1318, ack 1174, win 503, length 491: HTTP: HTTP/1.1 404 Not Found
04:56:40.756726 IP 167.179.68.23.53500 > 172.21.0.3.80: Flags [.], ack 1318, win 508, length 0
04:56:45.378128 IP 172.21.0.3.80 > 167.179.68.23.53500: Flags [F.], seq 1318, ack 1174, win 503, length 0
04:56:45.612988 IP 167.179.68.23.53500 > 172.21.0.3.80: Flags [.], ack 1319, win 508, length 0
04:57:11.289368 IP 172.21.0.3.80 > 167.179.68.23.53501: Flags [S.], seq 175704720, ack 2482258214, win 65320, options [mss 1420,nop,nop,sackOK,nop,wscale 7], length 0
04:57:11.523555 IP 167.179.68.23.53501 > 172.21.0.3.80: Flags [.], ack 1, win 514, options [nop,nop,sack 1 {0:1}], length 0
04:57:24.927366 IP 167.179.68.23.53501 > 172.21.0.3.80: Flags [.], seq 0:1, ack 1, win 514, length 1: HTTP
04:57:24.928033 IP 172.21.0.3.80 > 167.179.68.23.53501: Flags [.], ack 1, win 511, options [nop,nop,sack 1 {0:1}], length 0
04:57:30.616282 IP 167.179.68.23.53500 > 172.21.0.3.80: Flags [.], seq 1173:1174, ack 1319, win 508, length 1: HTTP
04:57:30.616818 IP 172.21.0.3.80 > 167.179.68.23.53500: Flags [.], ack 1174, win 503, length 0
```

访问Web-2：

```
chenman@mig-dm-secnet-ids-uscentral1-4rsz:~$ sudo tcpdump -i ens4 -nn -n "(icmp or port 80) and net 172.21.0.0/24"
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens4, link-type EN10MB (Ethernet), capture size 262144 bytes
04:58:27.389979 IP 167.179.68.23.53500 > 172.21.0.3.80: Flags [F.], seq 3408461424, ack 4078978408, win 508, length 0
04:58:27.390005 IP 167.179.68.23.53501 > 172.21.0.3.80: Flags [F.], seq 2482258214, ack 175704721, win 514, length 0
04:58:27.390276 IP 172.21.0.3.80 > 167.179.68.23.53500: Flags [R], seq 4078978408, win 0, length 0
04:58:27.390360 IP 172.21.0.3.80 > 167.179.68.23.53501: Flags [F.], seq 1, ack 1, win 511, length 0
04:58:27.622411 IP 167.179.68.23.53501 > 172.21.0.3.80: Flags [.], ack 2, win 514, length 0
04:58:32.758889 IP 167.179.68.23.53519 > 172.21.0.2.80: Flags [S], seq 865918889, win 64240, options [mss 1400,nop,wscale 8,nop,nop,sackOK], length 0
04:58:32.758915 IP 167.179.68.23.53520 > 172.21.0.2.80: Flags [S], seq 1614118438, win 64240, options [mss 1400,nop,wscale 8,nop,nop,sackOK], length 0
04:58:32.759234 IP 172.21.0.2.80 > 167.179.68.23.53519: Flags [S.], seq 191856946, ack 865918890, win 65320, options [mss 1420,nop,nop,sackOK,nop,wscale 7], length 0
04:58:32.759285 IP 172.21.0.2.80 > 167.179.68.23.53520: Flags [S.], seq 3704912818, ack 1614118439, win 65320, options [mss 1420,nop,nop,sackOK,nop,wscale 7], length 0
04:58:32.991336 IP 167.179.68.23.53519 > 172.21.0.2.80: Flags [.], ack 1, win 514, length 0
04:58:32.991458 IP 167.179.68.23.53520 > 172.21.0.2.80: Flags [.], ack 1, win 514, length 0
04:58:32.992639 IP 167.179.68.23.53520 > 172.21.0.2.80: Flags [P.], seq 1:433, ack 1, win 514, length 432: HTTP: GET / HTTP/1.1
04:58:32.993072 IP 172.21.0.2.80 > 167.179.68.23.53520: Flags [.], ack 433, win 507, length 0
04:58:32.993323 IP 172.21.0.2.80 > 167.179.68.23.53520: Flags [P.], seq 1:336, ack 433, win 507, length 335: HTTP: HTTP/1.1 200 OK
04:58:33.272694 IP 167.179.68.23.53520 > 172.21.0.2.80: Flags [.], ack 336, win 512, length 0
04:58:33.291768 IP 167.179.68.23.53520 > 172.21.0.2.80: Flags [P.], seq 433:800, ack 336, win 512, length 367: HTTP: GET /null HTTP/1.1
04:58:33.292104 IP 172.21.0.2.80 > 167.179.68.23.53520: Flags [P.], seq 336:827, ack 800, win 505, length 491: HTTP: HTTP/1.1 404 Not Found
04:58:33.531600 IP 167.179.68.23.53520 > 172.21.0.2.80: Flags [P.], seq 800:1174, ack 827, win 510, length 374: HTTP: GET /favicon.ico HTTP/1.1
04:58:33.531882 IP 172.21.0.2.80 > 167.179.68.23.53520: Flags [P.], seq 827:1318, ack 1174, win 503, length 491: HTTP: HTTP/1.1 404 Not Found
04:58:33.818868 IP 167.179.68.23.53520 > 172.21.0.2.80: Flags [.], ack 1318, win 508, length 0
04:58:38.440811 IP 172.21.0.2.80 > 167.179.68.23.53520: Flags [F.], seq 1318, ack 1174, win 503, length 0
```

测试Suricata Alter规则：

```
####RULES#####
#UDP ALERTS
alert udp $HOME_NET any -> 8.8.8.8 53 (msg:"BAD UDP DNS REQUEST"; sid:99996; rev:1;) egress

#HTTP ALERTS                                                                                    ingress
alert http any any -> $HOME_NET 80 (msg:"BAD HTTP PHP REQUEST"; http.uri; content:"index.php"; sid:99997; rev:1;)

#ICMP ALERTS
alert icmp any any -> $HOME_NET any (msg:"BAD ICMP"; sid:99998; rev:1;) ingress

#TCP ALERTS
alert tcp $HOME_NET any -> any 6667 (msg:"BAD TCP 6667 REQUEST"; sid:99999; rev:1;)   egress
~
```

测试Web Server 1，测试出向流量.DNS 解析，telnet 外部端口

```
dig @8.8.8.8 google.com

telnet 100.64.1.1 6667
```

在CLoud shell 中，测试packet mirrors subnet的入向流量：【ping Webserver、http访问】

```
ping -c 3  34.123.178.63

http://34.123.178.63/index.php
```