Secure Docker Network

With SWAG

SWAG - Secure Web Application Gateway

With iptables

?? Container ???????? iptables ???????????

- docker-compose: ?????????(br-*) ? `DOCKER-USER` ????
- docker: ?? `PRE_DOCKER` ????

for docker-compose)

```
#!/usr/bin/env bash

# Purpose: Restrict the access to the container from external IPs.
# Author: Alang, 2019/8/14
#

CONTAINER="nginx_mysql_web_1"
EXT_IF="eth0"
BRG_IF="$(ip a | awk '/br-[a-z0-9]+:/{print $0}' | awk '{print $2}' | sed 's/://g')"

if [ ! -x /usr/bin/docker ]; then
    echo "Abort: Unable to run the docker command!"
    echo "Recommendation: Check if the Docker was installed."
    exit 1
fi

if [ -z "$BRG_IF" ]; then
    echo "Abort: The network interface (br-????) with container not found. "
    exit 1
fi

# Check the network interfaces
```

```
for i in $EXT_IF $BRG_IF;do
    if ! (ip a show $i >/dev/null 2>&1); then
        echo "Abort: The network interface $i doesn't exist"
        exit 1
    fi
done

# Check if the DOCKER-USER chain exists, if it does flushing the rules.
iptables -C FORWARD -j DOCKER-USER 2> /dev/null

if [ $? -eq 0 ]; then
    # Flush all existing rules
    iptables -F DOCKER-USER
else
    echo "Abort: The chain DOCKER-USER not found!"
    echo "Recommendation: "
    echo "  - Is the Docker daemon running?"
    echo "  - Is the version of Docker 17.06+?"
    exit 1
fi

## Docker container named www-nginx public access policy
## If using docker-compose, change FORMAT to
## --format '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'
##
#WWW_IP="$(/usr/bin/docker inspect --format {{.NetworkSettings.IPAddress}} $CONTAINER 2>/dev/null)"
WWW_IP="$(/usr/bin/docker inspect --format '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'
$CONTAINER 2>/dev/null)"

if [ -z "$WWW_IP" ];then
    echo "Abort: The docker IP isn't detected!"
    echo "Recommendation:"
    echo "  - Is the specified container UP?"
    echo "  - Adjsut the variable \$WWW_IP."
    exit 1
fi

# Default action
iptables -I DOCKER-USER -i $EXT_IF -j DROP
```

```
## Insert web server container filter rules
## Allow All IPs to access 80 & 443
#iptables -I DOCKER-USER -i $EXT_IF -p tcp -d $WWW_IP --dport 80  -j ACCEPT
#iptables -I DOCKER-USER -i $EXT_IF -p tcp -d $WWW_IP --dport 443 -j ACCEPT
## Allow Cloudflare IPs to acces 443
## Check the IP list of Cloudflare at the URL https://www.cloudflare.com/ips-v4
iptables -I DOCKER-USER -o $BRG_IF -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
iptables -I DOCKER-USER -i $BRG_IF ! -o $BRG_IF -j ACCEPT
iptables -I DOCKER-USER -m state --state RELATED -j ACCEPT
iptables -I DOCKER-USER -i $BRG_IF -o $BRG_IF -j ACCEPT

iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 173.245.48.0/20  --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 103.21.244.0/22  --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 103.22.200.0/22  --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 103.31.4.0/22    --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 141.101.64.0/18  --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 108.162.192.0/18 --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 190.93.240.0/20  --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 188.114.96.0/20  --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 197.234.240.0/22 --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 198.41.128.0/17  --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 162.158.0.0/15   --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 104.16.0.0/12    --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 172.64.0.0/13    --dport 443 -j ACCEPT
iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 131.0.72.0/22    --dport 443 -j ACCEPT

# My IPs
#iptables -I DOCKER-USER -i $EXT_IF -p tcp  -d $WWW_IP -s 219.70.88.133    --dport 443 -j ACCEPT



[ $? -eq 0 ] && echo "Done."
## EOL
```

for docker)

```
#!/usr/bin/env bash


# Usage:
# timeout 10 docker_iptables.sh
#
```

```
# Use the builtin shell timeout utility to prevent infinite loop (see below)

CONTAINER="raida17"

if [ ! -x /usr/bin/docker ]; then
    exit
fi

# Check if the PRE_DOCKER chain exists, if it does there's an existing reference to it.
iptables -C FORWARD -o docker0 -j PRE_DOCKER 2> /dev/null

if [ $? -eq 0 ]; then
    # Remove reference (will be re-added again later in this script)
    iptables -D FORWARD -o docker0 -j PRE_DOCKER
    # Flush all existing rules
    iptables -F PRE_DOCKER
else
    # Create the PRE_DOCKER chain
    iptables -N PRE_DOCKER
fi

# Default action
iptables -I PRE_DOCKER -j DROP

# Docker Containers Public Admin access (insert your IPs here)
#iptables -I PRE_DOCKER -i eth0 -s 192.184.41.144 -j ACCEPT
#iptables -I PRE_DOCKER -i eth0 -s 120.29.76.14 -j ACCEPT

# Docker Containers Restricted LAN Access (insert your LAN IP range or multiple IPs here)
#iptables -I PRE_DOCKER -i eth1 -s 192.168.1.101 -j ACCEPT
#iptables -I PRE_DOCKER -i eth1 -s 192.168.1.102 -j ACCEPT

# Docker internal use
iptables -I PRE_DOCKER -o docker0 -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
iptables -I PRE_DOCKER -i docker0 ! -o docker0 -j ACCEPT
iptables -I PRE_DOCKER -m state --state RELATED -j ACCEPT
iptables -I PRE_DOCKER -i docker0 -o docker0 -j ACCEPT

# Docker container named www-nginx public access policy
# If using docker-compose, change format to
```

```
# --format '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'
WWW_IP_CMD="/usr/bin/docker inspect --format {{.NetworkSettings.IPAddress}} $CONTAINER"
WWW_IP=$($WWW_IP_CMD)

# Double check, wait for docker socket (upstart docker.conf already does this)
while [ ! -e "/var/run/docker.sock" ]; do echo "Waiting for /var/run/docker.sock..."; sleep 1; done

# Wait for docker web server container IP
while [ -z "$WWW_IP" ]; do echo "Waiting for $CONTAINER IP..."; WWW_IP=$($WWW_IP_CMD); done

## Insert web server container filter rules
## Allow All IPs to access 80 & 443
#iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP --dport 80  -j ACCEPT
#iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP --dport 443 -j ACCEPT
## Allow Cloudflare IPs to acces 443
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 103.21.244.0/22 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 103.22.200.0/22 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 103.31.4.0/22 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 104.16.0.0/12 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 108.162.192.0/18 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 131.0.72.0/22 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 141.101.64.0/18 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 162.158.0.0/15 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 172.64.0.0/13 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 173.245.48.0/20 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 188.114.96.0/20 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 190.93.240.0/20 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 197.234.240.0/22 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 198.41.128.0/17 --dport 443 -j ACCEPT
iptables -I PRE_DOCKER -i eth0 -p tcp -d $WWW_IP -s 199.27.128.0/21 --dport 443 -j ACCEPT


# Finally insert the PRE_DOCKER table before the DOCKER table in the FORWARD chain.
iptables -I FORWARD -o docker0 -j PRE_DOCKER
```

With iptables + ipset

- Limit Docker Container Access to Certain IP Addresses

With ufw

- ufw + Docker

With Firewalld

```
# Removing DOCKER-USER CHAIN (it won't exist at first)
firewall-cmd --permanent --direct --remove-chain ipv4 filter DOCKER-USER

# Flush rules from DOCKER-USER chain (again, these won't exist at first; firewalld seems to remember these even if the chain is gone)
firewall-cmd --permanent --direct --remove-rules ipv4 filter DOCKER-USER

# Add the DOCKER-USER chain to firewalld
firewall-cmd --permanent --direct --add-chain ipv4 filter DOCKER-USER

# Add rules (see comments for details)
firewall-cmd --permanent --direct --add-rule ipv4 filter DOCKER-USER 0 -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT -m comment --comment "This allows docker containers to connect to the outside world"
## Change the Docker Subnet address to your network settings (Could be 172.18.0.0/16)
firewall-cmd --permanent --direct --add-rule ipv4 filter DOCKER-USER 0 -j RETURN -s 172.17.0.0/16 -m comment --comment "allow internal docker communication"
## Add an ip to allow access to the docker exposed port
firewall-cmd --permanent --direct --add-rule ipv4 filter DOCKER-USER 0 -p tcp -m multiport --dports 9392 -s 10.18.109.20/32 -j ACCEPT -m comment --comment "my allowed ip address to openvas port"

# Add as many ip or other rules and then run this command to block all other traffic
firewall-cmd --permanent --direct --add-rule ipv4 filter DOCKER-USER 0 -j REJECT -m comment --comment "reject all other traffic"

# restart the services
systemctl stop docker
systemctl stop firewalld
systemctl start firewalld
systemctl start docker
```

Revision #10
Created 29 May 2020 07:06:30 by Admin
Updated 19 April 2022 06:54:37 by Admin