Gateway and Server Instructions

Overview

Consider this documentation a guide for how to set up your own security gateway that provides comparable functionality to the Helm service. If you are not familiar with how Helm works from a networking perspective - please read this first. This document is intentionally vague on some instructions. Running your own email server at home can be very challenging and risky. If you do not have sufficient experience to do so, we advise against doing this.

Note: Please remove the batteries from your Helm server before proceeding with these instructions.

Pre-requisites

- Static IPv4 address with good reputation
- A domain you own with corresponding DNS services, including the ability to create a PTR or Reverse DNS record
- An account with a cloud services provider like AWS, Azure, GCP, Linode, Digital Ocean, etc. Guidance in this document is oriented toward AWS as this was the platform Helm services were built on.

Allocate resources

On AWS, you will need to allocate an EC2 instance using Amazon Linux 2. A minimal instance like a T3a.nano will work fine. You will need to allocate an Elastic IP and assign it to this instance to ensure that its IP address is constant throughout its usage. Your PTR / Reverse DNS records should be configured with this IP address. Also, you will need to request that AWS remove email sending restrictions for this EC2 instance. We have heard mixed reports from individuals about AWS honoring these requests but it is necessary for traffic to flow through port 25.

Harden the instance

Take measures to harden the EC2 instance. Providing an exhaustive list of measures is beyond the scope of this document but some basic considerations include:

- disabling remote root login over ssh
- disabling password login over ssh
- removing unnecessary users
- ensuring only necessary ports are opened
- potentially restricting access to certain ports like the ssh port to a certain IP
- configuring automatic application of security updates

Configure the VPN connection

Strongswan can provide the IKEv2-based VPN service for forwarding traffic from the gateway to your server. You will need to install this on your gateway instance. We recommend reviewing the log levels to see what level of logging you are comfortable with for the balance between troubleshooting and privacy.

We have found the settings below in the sample ipsec.conf file are effective for creating durable and secure VPN connections.

```
# Here's an example ipsec.conf file
config setup
  charondebug="ike 3, knl 1, cfg 3"
  uniqueids=replace

conn box-vpn
  auto=add
  compress=no
  type=tunnel
  keyexchange=ikev2
  fragmentation=yes
  forceencaps=yes
  dpdaction=clear
  dpddelay=30s
  dpdtimeout=180s
  rekey=yes
  authby=secret
  ike=aes256gcm12-sha256-modp4096,aes128gcm12-sha256-modp4096!
  esp=aes256gcm12-sha256,aes128gcm12-sha256!
  leftsubnet=0.0.0.0/0
  leftupdown=/usr/bin/vpn-updown.sh
  right=%any
  rightsourceip=10.8.0.1/32
  leftid=@@EC2-GATEWAY
  rightid=@@BOX-HOME
  mobike=no
```

Configure iptables

iptables needs to be configured to route packets received on a particular port to the corresponding port on a vpn-connected client. The ports to forward for email are:

- 25 (sendmail port for email servers to communicate with each other)
- 587 (SMTP over TLS port)
- 993 (IMAP over SSL port)

Additional ports you may want to configure depending on services running on the server:

- 80, 443 (web traffic)
- 8443 (CalDAV/CardDAV)
- 9443 (Nextcloud - beware of high traffic costs for routing this traffic through your gateway)

Below are some reference commands that may be useful to configure this as needed.

```
# $local_ip - ip address for gateway as VPN peer
# $local_eth_dev - network interface on gateway to local network
# $box_virt_ip - ip address for server as VPN peer


#Clear out
iptables -t filter -F
```

```
iptables -t mangle -F
iptables -t nat -F

# --
iptables -t filter -P FORWARD DROP
iptables -t filter -A FORWARD -p icmp -j ACCEPT


#Setup Rules for World->Gateway->Box
iptables -t filter -A FORWARD -d $box_virt_ip -m conntrack --ctstate
RELATED,ESTABLISHED -j ACCEPT

setup_forwarding_port () {
  port=$1

  echo "  ... PREROUTING($port)"
  iptables -t nat -A PREROUTING \
              -i $local_eth_dev -d $local_ip \
              -p tcp -m tcp \
              --dport $port -j DNAT \
              --to-destination $box_virt_ip

  echo "  ... FORWARD($port)"
  iptables -t filter -A FORWARD \
             -i $local_eth_dev -d $box_virt_ip \
             -p tcp -m tcp --dport $port \
             --tcp-flags FIN,SYN,RST,ACK SYN \
             -m conntrack --ctstate NEW -j ACCEPT
}

ports=(80 443 25 993 587 3333 3334 8443)
for port in ${ports[@]}; do
  echo "Setting up port for $port"
  setup_forwarding_port $port
done

#Setup Rules for Box->Us->World
iptables -t filter -A FORWARD -i $local_eth_dev -s $box_virt_ip -j ACCEPT
iptables -t filter -A FORWARD -s $box_virt_ip/32 -o $local_eth_dev -m
policy --dir out --pol ipsec -j ACCEPT
iptables -t filter -A FORWARD -d $box_virt_ip/32 -o $local_eth_dev -m
policy --dir out --pol ipsec -j ACCEPT
iptables -t nat -A POSTROUTING -s $box_virt_ip/32 -o $local_eth_dev -j
MASQUERADE


iptables -t mangle -A FORWARD -p tcp -m tcp --tcp-flags SYN,RST SYN -m
tcpmss --mss 1361:65535 -j TCPMSS --set-mss 1360
```

Server Instructions

Now that the gateway has been configured and is listening for connections, the server can be configured. These instructions will assist with configuring the server as a VPN client to connect to the gateway so traffic can be forwarded on the necessary ports.

VPN Connection

```
n_user_vip_ifcs=$(ip addr show |grep -o "$USER_VPN_SERVER_VIRTUAL_IP" |wc -
l)
wired_up=$(ip addr show wlP2p1s0 |grep inet | grep -v
$USER_VPN_SERVER_VIRTUAL_IP |grep -v 10.8.0.2)
wifi_up=$(ip addr show eth0 |grep inet | grep -v
$USER_VPN_SERVER_VIRTUAL_IP |grep -v 10.8.0.2)

if [ -n "$wired_up" ]; then
  log-helper info "Setting up $USER_VPN_SERVER_VIRTUAL_IP on eth0"
  ip addr add $USER_VPN_SERVER_VIRTUAL_IP/32 dev eth0
  ip addr del $USER_VPN_SERVER_VIRTUAL_IP/32 dev wlP2p1s0
elif [ -n "$wifi_up" ]; then
  log-helper info "Setting up $USER_VPN_SERVER_VIRTUAL_IP on wlP2p1s0"
  ip addr add $USER_VPN_SERVER_VIRTUAL_IP/32 dev wlP2p1s0
  ip addr del $USER_VPN_SERVER_VIRTUAL_IP/32 dev eth0
else
  log-helper info "Setting up $USER_VPN_SERVER_VIRTUAL_IP on eth0"
  ip addr add $USER_VPN_SERVER_VIRTUAL_IP/32 dev eth0
  ip addr del $USER_VPN_SERVER_VIRTUAL_IP/32 dev wlP2p1s0
fi
set -e # Stop ignoring errors

log-helper info "Setting up gateway ip..."
if [ -n "$GATEWAY_IP" ]; then
  sed -i "s/ right=.*/right=$GATEWAY_IP/" /etc/ipsec.
conf
else
  log-helper error "GATEWAY_IP must be provided"
fi

#If we use a private self signed cacert & serevr cert do this
#log-helper info "Configuring local certs..."
#/usr/bin/setup-vpn-certs.sh $DOMAIN

log-helper info "Setting up keyfile..."
if [ ! -n "$STATIC_KEY" ]; then
  log-helper info "using default built in KEY"
  cp ${CONTAINER_SERVICE_DIR}/strongswan/assets/static.key /tmp/
else
  # strip carraige returns out
  echo -n "$STATIC_KEY" > /tmp/static.key
fi



log-helper info "Setting up tunnel bypasses..."
if [ -n "$BYPASSES" ]; then
```

```
  # setup user specified bypasses
  # bypass list is in format "plabs-20-net=172.x.x.x/8,home-net=192.168.0.0
/16" etc

  bypass_list=$(echo "$BYPASSES" | tr ',' '\n')
  fname="/etc/ipsec.conf"

  while read net; do
    name=$(echo "$net" | sed 's/=.*//')
    ip_range=$(echo "$net" | sed 's/.*=//')

    log-helper info "Adding VPN bypass '$name' = '$ip_range'"

    echo "" >> $fname
    echo "conn $name" >> $fname
    echo "    leftsubnet=$ip_range" >> $fname
    echo "    rightsubnet=$ip_range" >> $fname
    echo "    authby=never" >> $fname
    echo "    type=passthrough" >> $fname
    echo "    auto=route" >> $fname
    echo "" >> $fname

  done <<< "$bypass_list"

fi


PSK_KEY="$(cat /tmp/static.key)"

log-helper info "Setting up ipsec.secrets/ipsec.conf"
rm -rf /etc/ipsec.secrets
touch /etc/ipsec.secrets
chmod 600 /etc/ipsec.secrets
echo " @@EC2-GATEWAY : PSK \"${PSK_KEY}\" " >> /etc/ipsec.secrets

mkdir -p /data/strongswan/etc
echo "${VPN_SUBDOMAIN_NAME}.${DOMAIN}" > /data/strongswan/etc/domain.txt

# Save domain for later -- will be used by process.sh/maintain_vpn_cert.sh

log-helper info "Setting up user connections for ipsec.conf"
fill_in_user_vpn_ip_values $ASSETS/ipsec.user_device.conf

cat $ASSETS/ipsec.user_device.conf >> /etc/ipsec.conf
sed -i "s/{{MY_DOMAIN}}/${VPN_SUBDOMAIN_NAME}.$DOMAIN/" /etc/ipsec.conf

echo " : RSA /etc/ipsec.d/private/server-vpn-key.pem" >> /etc/ipsec.secrets

log-helper info "Setting radius/ldap connection..."

sed -i "s/SECRET/$radius_password/" /etc/strongswan.conf
```

Services

Here are some pointers to documentation and open source projects that may provide services similar to Helm for email, calendar, contacts and file sharing.

Documentation for setting up your own mail stack:

- https://www.linode.com/docs/guides/email-with-postfix-dovecot-and-mysql/
- https://www.linuxbabe.com/mail-server/setup-basic-postfix-mail-sever-ubuntu

Pre-existing open source projects that provide an integrated mail stack:

- https://mailinabox.email/
- https://github.com/sovereign/sovereign

File sync/share:

- https://docs.nextcloud.com/server/latest/admin_manual/installation/index.html

Troubleshooting Tips

Networking

Ensure that your home networking equipment is not blocking outbound traffic on ports 500 and 4500 which are used for the VPN connection.

Email Authentication

- Understanding how DKIM, DMARC and SPF work together - https://www.techtarget.com/searchsecurity/answer/Email-authentication-How-SPF-DKIM-and-DMARC-work-together