Private IP addresses and NAT

Networking with jails

November 26, 2019

The easiest way to provide jails with a network address is to use the parameter `ip4.addr`.

```
jail01 {
   ip4.addr = "em0|10.0.2.80";
}
```

This downside of this method is that it requires fixed IP addresses on your network segment. In an enterprise this requires communication with the networking team, in the cloud this means paying money for additional public IP addresses. This article helps you save money as well as helping you avoid interacting with the dreaded folks of the networking team.

The following steps are required to get this setup working:

1. Allocate a range of private IP addresses to your jails. In the case of a cloud server you can choose any RFC 1918 address range as your routable IP address is a public IP address. Should you want to use this method on a server within your enterprise—which uses RFC 1918 addresses—you must pick a unique range to avoid overlapping IP addresses. Either align with your networking team so they can reserve a block of IP addresses for your jails, or pick a range that should not be used anywhere else, e.g. the test-net addresses from RFC 5735.

2. Create a new loopback interface on the host and assign it an IP address from this range. It will act as the default gateway for your jails and perform network address translation (NAT) on behalf of the jails.

3. Create the necessary firewall rules for network address translation on the host.
4. Create your jails.

Let's revisit these actions in more detail and add some code.

IP address space

In this article I will use one of the test-net ranges (203.0.113.0/24) from RFC 5735 as these addresses won't conflict with neither the public IP space on the Internet nor with the routable IP addresses of the enterprise.

If you ask the company's networking team for a unique range of IP addresses, you can also opt for not using network address translation. However, this requires the networking team to route traffic for the given IP range to your FreeBSD host, something they might not want to do.

Loopback interface

Edit */etc/rc.conf* and add the following lines, creating a new loopback interface.

```
cloned_interfaces="lo100"
ifconfig_lo100="203.0.113.1/24"
```

Next create the interface; this step will be done automatically at startup the next time you reboot the server.

```
# ifconfig lo100 create
```

Firewall rules

In this article I will use PF firewall rules. The configuration for the IPFW and IPF firewalls will be similar.

We will start with allowing outbound access for all the jails. First create two macros to make the configuration more readable.

```
INTF_PUB="em0"
NET_JAILS="203.0.113.0/24"
INTF_JAILS="lo100"
```

Next exclude inter-jail communication. It might not be a good idea to do this, but that would depend on your setup in which case you should just create firewall rules for each and every jail on the loopback interface.

```
set skip on { lo0 $INTF_JAILS }
```

Finally create a NAT rule to translate outbound traffic.

```
nat pass on $INTF_PUB from $NET_JAILS to any -> ($INT_PUB)
```

See the man page on where to put these statements and they have a strict order.

Create jails

Now create the jail and add its configuration to */etc/jail.conf*, e.g.:

```
nattest {
   ip4.addr="203.0.113.80";
   allow.raw_sockets;
}
```

The second statement is for testing purposes only and should be removed in production environments for added security.

If the jail requires inbound access, add the necessary rules to */etc/pf.conf*:

```
rdr on $INTF_PUB proto tcp \
   from any to ($INTF_PUB) port 80 \
   -> 203.0.113.80 port 80
```