네트워크 네임스페이스 (최영락, 오픈스택 스터디)

2016년 4월 22일 금요일

네임스페이스 (namespace)는 한글로 "이름 공간"이라고 하는데, 한 Linux 운영체제 내에서 사용하는 네트워크에 대한 영역을 논리적으로 구분시켰다고 보면 된다. 네트워크 네임스페이스 자체에 대한 자세한 내용은 http://ko.sdndev.net/2 를 참고하자.

http://www.opencloudblog.com/?p=116 블로그에 있는 내용을 실습해 본 내용을 블로그에 담아보았다.

- 네임스페이스 생성: ns1과 ns2라는 이름을 가진 네임스페이스를 추가하자.
주의할 사항은 반드시 root 권한에서 실행해야 한다는 것이다. 일반 사용자 권한으로 실행하면 아래와 같은 결과를 얻는다.

```
ian@ianychoi:~$ ip netns add ns1
mount --make-shared /var/run/netns failed: No such file or directory
ian@ianychoi:~$
```

두 개의 네임스페이스를 추가한 결과이다. 네임스페이스 목록은 "ip netns" 명령어를 실행하여 확인 가능하다.

```
root@ianychoi:~# ip netns add ns1
root@ianychoi:~# ip netns add ns2
root@ianychoi:~#
root@ianychoi:~# ip netns
ns2
ns1
root@ianychoi:~#
```

생성된 네임스페이스 목록은 /var/run/netns/ 폴더에서도 확인 가능하다.

```
root@ianychoi:~# ls -al /var/run/netns
total 0
drwxr-xr-x  2 root root   80 Apr 22 02:02 .
drwxr-xr-x 30 root root 1020 Apr 22 02:02 ..
-r--r--r--  1 root root    0 Apr 22 02:02 ns1
-r--r--r--  1 root root    0 Apr 22 02:02 ns2
```

두 개의 네임스페이스에 대해 root 권한으로 생성되었음을 확인할 수 있다.
참고로 Linux에서는 일반적으로 현재 실행 중인 리소스들을 /var/run 내에 파일로 위치시킨다.

이 네임스페이스는 네트워크에 대한 부분을 각 네임스페이스별로 격리시킨다.
네임스페이스를 생성하기 전, 즉 현재 사용하고 있는 Linux 환경은 기본 네임스페이스로 본다.
/proc 내에는 Linux에서 실제 사용 중인 여러 자원에 대한 정보를 파일로 살펴볼 수 있도록 관리하고 있다.
/proc/self/ns 에는 현재 자신의 네임스페이스에 대한 내용을 볼 수 있으며, 이 중 net은 네트워크 네임스페이스를 의미한다.
참고로 Linux 파일 시스템에서는 모든 파일에 대한 자원을 inode라는 고유한 번호로 관리하고 있다.
그리고 다른 네트워크 네임스페이스에 있는 항목을 살펴보려면 "ip netns exec [네임스페이스 이름]

[명령어]" 형식으로 실행한다.

```
root@ianychoi:~# ls -al /proc/self/ns
total 0
dr-x--x--x 2 root root 0 Apr 22 02:18 .
dr-xr-xr-x 9 root root 0 Apr 22 02:18 ..
lrwxrwxrwx 1 root root 0 Apr 22 02:18 ipc -> ipc:[4026531839]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 mnt -> mnt:[4026531840]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 net -> net:[4026531956]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 pid -> pid:[4026531836]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 user -> user:[4026531837]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 uts -> uts:[4026531838]
root@ianychoi:~# ip netns exec ns1 ls -al /proc/self/ns
total 0
dr-x--x--x 2 root root 0 Apr 22 02:18 .
dr-xr-xr-x 9 root root 0 Apr 22 02:18 ..
lrwxrwxrwx 1 root root 0 Apr 22 02:18 ipc -> ipc:[4026531839]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 mnt -> mnt:[4026532436]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 net -> net:[4026532109]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 pid -> pid:[4026531836]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 user -> user:[4026531837]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 uts -> uts:[4026531838]
root@ianychoi:~# ip netns exec ns2 ls -al /proc/self/ns
total 0
dr-x--x--x 2 root root 0 Apr 22 02:18 .
dr-xr-xr-x 9 root root 0 Apr 22 02:18 ..
lrwxrwxrwx 1 root root 0 Apr 22 02:18 ipc -> ipc:[4026531839]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 mnt -> mnt:[4026532436]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 net -> net:[4026532328]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 pid -> pid:[4026531836]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 user -> user:[4026531837]
lrwxrwxrwx 1 root root 0 Apr 22 02:18 uts -> uts:[4026531838]
root@ianychoi:~#
```

실행 결과를 보면 다음과 같은 특징을 알 수가 있다.

- ipc, pid, user, uts: 모두 동일
- net: 모두 다름
- mnt: 기본 네임스페이스에서만 다르고, 각 네임스페이스에서는 같다.

여기서 mnt는 실제 프로세스 실행에 의해 달라지는 것으로 보인다.

아래 listns.py는 http://www.opencloudblog.com/?p=251 에서 다운로드하였다.

```
root@ianychoi:~# python listns.py | grep mnt
       145  mnt:[4026531856]      kdevtmpfs
root@ianychoi:~# ip netns exec ns1 python listns.py | grep mnt
       145  mnt:[4026531856]      kdevtmpfs
     46639  mnt:[4026532436]      python listns.py
root@ianychoi:~# ip netns exec ns2 python listns.py | grep mnt
       145  mnt:[4026531856]      kdevtmpfs
     46641  mnt:[4026532436]      python listns.py
root@ianychoi:~#
```

현재 마운트된 네임스페이스는 다음 명령어를 통해서도 확인 가능하다.

```
root@ianychoi:~# cat /proc/self/mounts | grep -E "netns|ns1|ns2"
tmpfs /run/netns tmpfs rw,nosuid,noexec,relatime,size=713784k,mode=755 0 0
proc /run/netns/ns1 proc rw,nosuid,nodev,noexec,relatime 0 0
proc /run/netns/ns2 proc rw,nosuid,nodev,noexec,relatime 0 0
root@ianychoi:~# ip netns exec ns1 cat /proc/self/mounts | grep -E "netns|ns1|ns
2"
tmpfs /run/netns tmpfs rw,nosuid,noexec,relatime,size=713784k,mode=755 0 0
proc /run/netns/ns1 proc rw,nosuid,nodev,noexec,relatime 0 0
proc /run/netns/ns2 proc rw,nosuid,nodev,noexec,relatime 0 0
ns1 /sys sysfs rw,relatime 0 0
root@ianychoi:~# ip netns exec ns2 cat /proc/self/mounts | grep -E "netns|ns1|ns
2"
tmpfs /run/netns tmpfs rw,nosuid,noexec,relatime,size=713784k,mode=755 0 0
proc /run/netns/ns1 proc rw,nosuid,nodev,noexec,relatime 0 0
proc /run/netns/ns2 proc rw,nosuid,nodev,noexec,relatime 0 0
ns2 /sys sysfs rw,relatime 0 0
root@ianychoi:~#
```

그리고 두 네임스페이스를 연결하는 인터페이스를 추가해 보자.
먼저 첫 번째 명령어를 실행해 보자.

```
root@ianychoi:~# ip link add veth-a type veth peer name veth-b
root@ianychoi:~#
```

결과가 아무것도 나타나지 않았으나, 실제로 veth-a 와 veth-b 이름으로 가상 네트워크 인터페이스를 생성한 것이다. 그냥 "ip link add type veth"로도 생성할 수 있는데 이 때는 번호 순서대로 생성된다.
즉, veth가 아무것도 없을 경우 veth0과 veth1 으로 생성된다.

```
root@ianychoi:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group defa
ult qlen 1000
    link/ether 00:0d:3a:40:35:dc brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.4/24 brd 10.0.0.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20d:3aff:fe40:35dc/64 scope link
       valid_lft forever preferred_lft forever
5: veth-b: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default ql
en 1000
    link/ether 96:15:5d:29:70:d2 brd ff:ff:ff:ff:ff:ff
6: veth-a: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default ql
en 1000
    link/ether de:a3:36:d1:ad:64 brd ff:ff:ff:ff:ff:ff
root@ianychoi:~#
```

그 다음 명령어를 실행하자.

```
root@ianychoi:~# ip link set veth-a netns ns1
root@ianychoi:~#
```

역시 아무것도 나타나지 않는다. 그런데, 이 때 "ip a" 명령어를 실행하면 veth-a가 사라졌다!

veth-a는 ns1 네트워크 네임스페이스에 할당되었기 때문에 ns1 네임스페이스로 들어가야만 확인할 수 있다.

```
root@ianychoi:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group defa
ult qlen 1000
    link/ether 00:0d:3a:40:35:dc brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.4/24 brd 10.0.0.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20d:3aff:fe40:35dc/64 scope link
       valid_lft forever preferred_lft forever
5: veth-b: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default ql
en 1000
    link/ether 96:15:5d:29:70:d2 brd ff:ff:ff:ff:ff:ff
root@ianychoi:~# ip netns exec ns1 ip a
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
6: veth-a: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default ql
en 1000
    link/ether de:a3:36:d1:ad:64 brd ff:ff:ff:ff:ff:ff
root@ianychoi:~#
```

즉, 네트워크 네임스페이스는 Linux의 네트워크 인터페이스를 해당 네임스페이스에서만 사용할 수 있도록 별도로 격리하여 관리하는 체계에 해당한다.

그 다음 명령어를 실행하면 veth-b 인터페이스를 ns2에 위치시킨다.

```
root@ianychoi:~# ip link set veth-b netns ns2
root@ianychoi:~#
```

```
root@ianychoi:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group defa
ult qlen 1000
    link/ether 00:0d:3a:40:35:dc brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.4/24 brd 10.0.0.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20d:3aff:fe40:35dc/64 scope link
       valid_lft forever preferred_lft forever
root@ianychoi:~# ip netns exec ns1 ip a
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
6: veth-a: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default ql
en 1000
    link/ether de:a3:36:d1:ad:64 brd ff:ff:ff:ff:ff:ff
root@ianychoi:~# ip netns exec ns2 ip a
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
5: veth-b: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default ql
en 1000
    link/ether 96:15:5d:29:70:d2 brd ff:ff:ff:ff:ff:ff
root@ianychoi:~#
```

이 때부터 주의할 것은 veth-a와 veth-b는 더 이상, "ip netns exec ns1 [명령어]", "ip netns exec ns2 [명령어]"를 통해서는 확인할 수 없다는 것이다. 우선 방금 eth-b를 ns2로 이동한 직후, 같은 명령어를 다시 실행하면 veth-b를 찾을 수 없다고 한다.

```
root@ianychoi:~# ip link set veth-b netns ns2
root@ianychoi:~# ip link set veth-b netns ns2
Cannot find device "veth-b"
root@ianychoi:~#
```

이제 veth-b는 ns2 네임스페이스에 속해있기 때문이다.

블로그에는 그 다음 명령어로 PID 1234에 붙이는 예시를 설명하고 있는데, 즉 네트워크 네임스페이스를 실행하는 프로세스에 연결시킬 수 있음을 의미한다. 따라서 실행하는 프로세스 목록 중 어떤 프로세스는 ns1 네트워크 네임스페이스에 있는 네트워크 인터페이스를 사용하고, 또 다른 어떤 프로세스는 ns2 네트워크 네임스페이스에 있는 다른 네트워크 인터페이스를 사용할 수 있다는 점이다.