NISTNET: A Network Emulator for Protocol Experimentation

2005년 8월

하종식, 고석주

mugal1@cs.knu.ac.kr

요 약

NISTNET 은 Linux 위에서 동작하는 network emulation package 이다. 이 package 를 이용하여 우리는 다양한 network 을 하나의 Linux router 상에 구성할 수 있다. 일반적으로 network 은 packet loss, duplication, delay and jitter, bandwidth limitations, and network congestion 와 같은 상태를 가질 수 있는데 NISTNET 는 아래와 같은 real network 과 유사한 환경을 제공하게 한다. 본 문서에서는 NISTNET 의 설치 방법과 설치 후 packet loss, duplication, delay 값 변경에 대한 Testing 결과를 정리한다.

Table of Contents

1. What is NISTNET?

NISTNET 은 Linux위에서 동작하는 network emulation package이다. 이 package를 이용하여 우리는 넓고 다양한 network 상태를 하나의 Linux router를 통해서 구성할 수 있다. 일반적으로 network 은 packet loss, duplication, delay and jitter, bandwidth limitations, and network congestion 와 같은 상태를 가질 수 있는데 NISTNET는 아래와 같은 real network과 유사한 환경을 제공하게 한다.

① 실제 Network에서의 특징들

1. 바쁘고 혼잡한 IP network 과 datagram의 loss로 인한 수 많은 error들
2. IP 위에서의 느린 data이동, satellite와 같이 먼 거리의 환경에서의 high latency
3. high bandwidth LAN은 낮은 bandwidth link를 가진다.

② What is "emulation," and how does it differ from "simulation"?

"emulation"은 기본적으로 testing code가 "live" implementation 상황에서 동작을 한다, (allowing the live implementation to emulate (imitate) the performance characteristics of other networks.) "Simulation"은 전체적인 live한 점이 없는 synthetic test environment이다.

③ NISTNET and Client Software

NISTNET은 Linux 운영체제에 kernel module로서 구현되었다. 그리고 실제적으로 kernel과 interface의 역할을 하는 xnistnet이라는 software가 존재한다. xnistnet은 일반사용자도 쉽게 사용할 수 있도록 visual한 환경을 제공하는데 이를 사용하기 위해서는 X Window System이 깔려 있어야 한다.

2. Setup Guide

① Finding

http://snad.ncsl.nist.gov/itg/nistnet/install.html site에가면 소스를 구할 수 있다.

nistnet.2.0.12b.tar.gz (for 2.0 - 2.4 kernels) 와 nistnet.2.0.12c.tar.gz (for 2.6 kernels).

각 kernel version에 맞게 다운받으시길 바란다.

② Compiling and Install

NISTNET을 Linux 위에 올리기 위해서는 kernel 패치를 통해 하는 방법과 module로써 올리는 두 가지 방법이 존재하는데 일반적으로 두 번째 방법을 많이 이용하기에 본 리포트에서는 두 번째 방법을 이용하였다.

본인의 kernel version은 kernel-2.4 version이다

1. 먼저 kernel 2.0 - 2.4위에서 동작하는 nistnet.2.0.12b.tar.gz 다운 받는다.
2. tar -zxvf nistnet.2.0.12b.tar.gz 명령어를 통해 압축을 푼다.
3. cd nistnet.2.0.12b로 이동 후
4. ./configure를 수행한다. 그러면 아래와 같은 message들이 나올것이다.

- Decide whether or not you want support for explicit congestion notification processing. Also, decide whether you want COS (class of service) selection support.
- may want to use an alternative to the
- Tested with Athena Xaw, Xaw3d, and neXtaw

Build and install the nistnet module, API library, and user interface

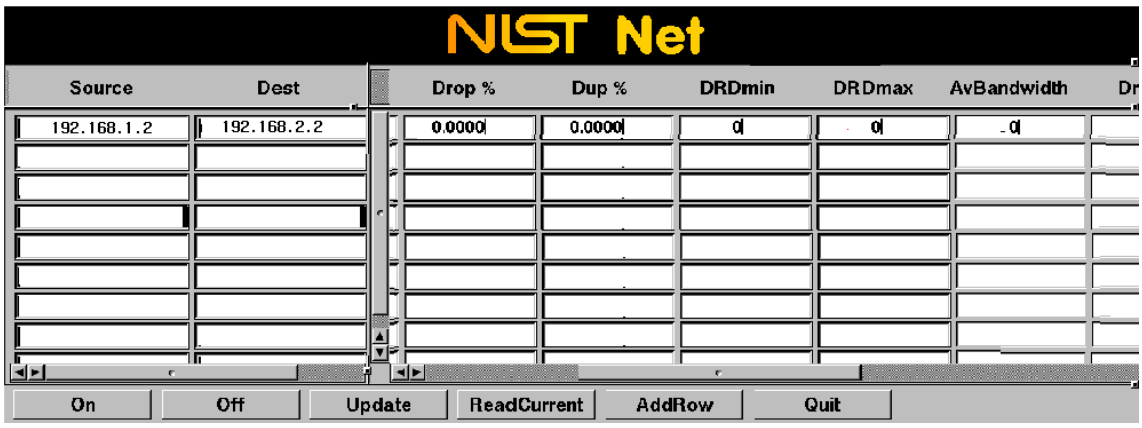
5. make
6. make install
7. Load.Nistnet LISTNET kernel module를 kernel에 올리는 명령이다 정상적으로 module이 올라가면 성공했다는 message를 볼 수 있다.

위와 같은 과정을 수행했을 시에 kernel module이 load된다면 lsmod를 통해서 확인 할 수 있다. 별 문제 없이 kernel이 잘 로드 됐다면 기본적으로 NISTNET을 사용하기 위한 준비는 되었다.

③ Using nistnet

위의 과정에서 nistnet을 사용하기 위한 kernel 설정이 완료 되었다 그럼 우리는 nistnet client를 이용하여 내가 원하는 network 환경을 구성할 수 있다. nistnet을 사용하기 위한 client는 두 가지가 있는데 하나는 xnistnet 이고 다른 하나는 cnistnet이다. 둘 다 하는 기능은 동일 하나 xnistnet 그래픽 interface를 제공하고 cnistnet 명령 행 방식이다.

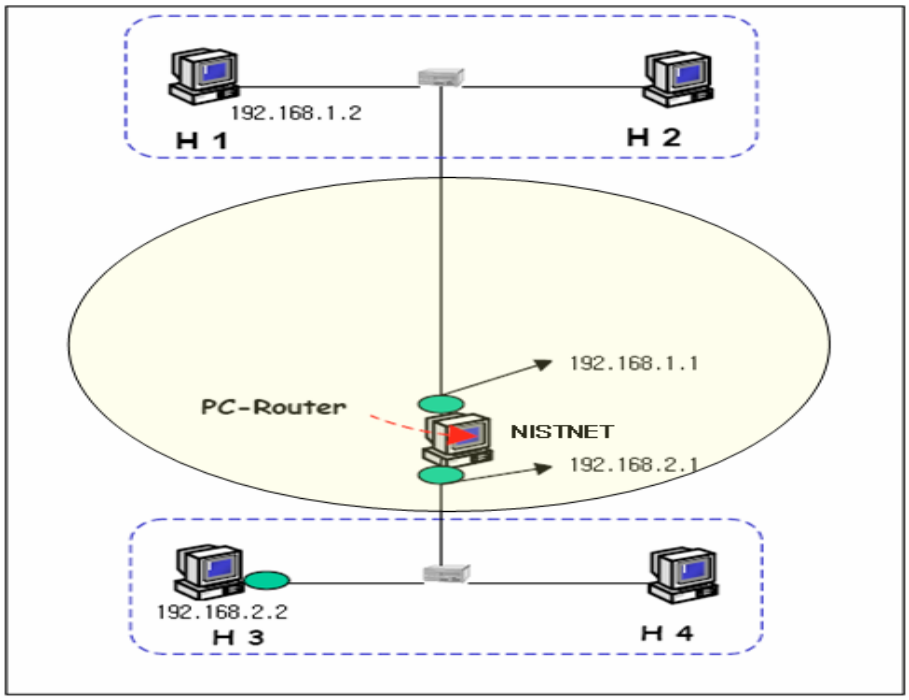
본 리포트에서는 일반적으로 가장 쉽고 많이 사용하는 xnistnet에 대해서 설명할 것이다. xnistnet이라고 명령하면 아래와 같은 프로그램이 동작 할 것 이다. 아래와 같은 프로그램이 동작한다면 정상적으로 xnistnet설치를 완료 한 것이다.



<XNISTNET 실행화면>

3. Testbed Configuration

아래 그림은 본 연구실에 설치된 NISTNET을 이용한 Emulating Network 구성도이다. 본 시스템은 3대의 Linux system(H1,H3,NISTET PC-Router)이 탑재된 컴퓨터를 가진다.



<연구실 NISTNET 구성도>

NISTET PC-Router는 두 개의 interface card를 가지고 192.168.1.0과 192.168.2.0의 두 개의 subnet을 구성하였다. 192.168.1.0 network에 속한 H1에는 192.168.1.2 IP Address를 할당 하였고 192.168.2.0 network에 속한 H2에는 192.168.1.2 IP Address를 할당 하였다. 본 실험에서는 H1에서 H2쪽으로 data를 보내고, nistnet의 설정에 따른 변경 사항들을 살펴 볼 것이다.

4. Testing

① Stable State

NESTNET을 작동시키지 않고 ping test를 한 상황이다. 결과는 아래와 같다.

```
1  PING 192.168.2.2 (192.168.2.2) 1460(1488) bytes of data.
2  1468 bytes from 192.168.2.2: icmp_seq=1 ttl=127 time=1.29 ms
3  1468 bytes from 192.168.2.2: icmp_seq=2 ttl=127 time=1.26 ms
4  1468 bytes from 192.168.2.2: icmp_seq=3 ttl=127 time=1.25 ms
5  1468 bytes from 192.168.2.2: icmp_seq=4 ttl=127 time=1.25 ms
6  1468 bytes from 192.168.2.2: icmp_seq=5 ttl=127 time=1.25 ms
7  1468 bytes from 192.168.2.2: icmp_seq=6 ttl=127 time=1.24 ms
8  1468 bytes from 192.168.2.2: icmp_seq=7 ttl=127 time=1.25 ms
9  1468 bytes from 192.168.2.2: icmp_seq=8 ttl=127 time=1.26 ms
10 1468 bytes from 192.168.2.2: icmp_seq=9 ttl=127 time=1.25 ms
11 1468 bytes from 192.168.2.2: icmp_seq=10 ttl=127 time=1.25 ms
12
13 --- 192.168.2.2 ping statistics ---
14 10 packets transmitted, 10 received, 0% packet loss, time 18006ms
15 rtt min/avg/max/mdev = 1.245/1.258/1.297/0.034 ms
```

② Drop packet loss late

특정한 상황 변수에 맞게 network을 emulation 하기 위해서 xnistnet을 실행시키고 source address를 192.168.1.2로 destination address를 192.168.2.2로 설정을 한 후 Random하게 packet을 drop 할 때의 ping test를 실행하였다. 본 실험에서는 NISTNET emulate에 30% packet loss를 발생시키게 설정을 하고 다른 delay, dup 와 같은 field 들은 0으로 setting하여 순수한 packet drop에 의 한 performance를 측정해 보았다. 결론적으로 평균 packet loss late이 30%정도가 된다는 것을 확인 할 수 있다.

```
PING 192.168.2.2 (192.168.2.2) 1460(1488) bytes of data.
1468 bytes from 192.168.2.2: icmp_seq=2 ttl=127 time=1.25 ms
1468 bytes from 192.168.2.2: icmp_seq=3 ttl=127 time=1.27 ms
1468 bytes from 192.168.2.2: icmp_seq=4 ttl=127 time=1.26 ms
1468 bytes from 192.168.2.2: icmp_seq=5 ttl=127 time=1.25 ms
1468 bytes from 192.168.2.2: icmp_seq=8 ttl=127 time=1.25 ms
1468 bytes from 192.168.2.2: icmp_seq=9 ttl=127 time=1.26 ms
1468 bytes from 192.168.2.2: icmp_seq=10 ttl=127 time=1.27 ms

--- 192.168.2.2 ping statistics ---
10 packets transmitted, 7 received, 30% packet loss, time 18002ms
rtt min/avg/max/mdev = 1.254/1.265/1.279/0.021 ms
```

③ Delay

본 실험에서는 NISTNET emulate에 한 packet 당 600 millisecond delay를 발생시키게 설정을 하고 다른 drop packet, dup 와 같은 field 들은 0으로 setting하여 순수한 packet delay에 의 한 performance를 측정해 보았다. 측정결과 하나의 packet에 대해 delay time이 일반적인 상황에서 보다 600ms씩 증가한 것을 볼 수 있다.

```
PING 192.168.2.2 (192.168.2.2) 1460(1488) bytes of data.
1468 bytes from 192.168.2.2: icmp_seq=1 ttl=127 time=601 ms
1468 bytes from 192.168.2.2: icmp_seq=2 ttl=127 time=601 ms
1468 bytes from 192.168.2.2: icmp_seq=3 ttl=127 time=601 ms
1468 bytes from 192.168.2.2: icmp_seq=4 ttl=127 time=601 ms
1468 bytes from 192.168.2.2: icmp_seq=5 ttl=127 time=601 ms
1468 bytes from 192.168.2.2: icmp_seq=6 ttl=127 time=601 ms
1468 bytes from 192.168.2.2: icmp_seq=7 ttl=127 time=601 ms
1468 bytes from 192.168.2.2: icmp_seq=8 ttl=127 time=601 ms
1468 bytes from 192.168.2.2: icmp_seq=9 ttl=127 time=601 ms
1468 bytes from 192.168.2.2: icmp_seq=10 ttl=127 time=601 ms

--- 192.168.2.2 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 17996ms
rtt min/avg/max/mdev = 601.242/601.377/601.946/0.397 ms
```

④ Duplicate

본 실험에서는 NISTNET emulate에 한 packet 당 30% duplicate 를 발생시키게 설정을 하고 다른 drop packet, delay 와 같은 field 들은 0으로 setting하여 순수한 dup 에 의 한 performance를 측정해 보았다. 측정결과 10개중 2개의 duplicate가 발생한 것을 확인 할 수 있었다.

```
 1  PING 192.168.2.2 (192.168.2.2) 1460(1488) bytes of data.
 2  1468 bytes from 192.168.2.2: icmp_seq=1 ttl=127 time=1.42 ms
 3  1468 bytes from 192.168.2.2: icmp_seq=2 ttl=127 time=1.27 ms
 4  1468 bytes from 192.168.2.2: icmp_seq=3 ttl=127 time=1.26 ms
 5  1468 bytes from 192.168.2.2: icmp_seq=3 ttl=127 time=1.41 ms (DUP!)
 6  1468 bytes from 192.168.2.2: icmp_seq=4 ttl=127 time=1.26 ms
 7  1468 bytes from 192.168.2.2: icmp_seq=5 ttl=127 time=1.25 ms
 8  1468 bytes from 192.168.2.2: icmp_seq=6 ttl=127 time=1.26 ms
 9  1468 bytes from 192.168.2.2: icmp_seq=6 ttl=127 time=1.41 ms (DUP!)
10  1468 bytes from 192.168.2.2: icmp_seq=7 ttl=127 time=1.26 ms
11  1468 bytes from 192.168.2.2: icmp_seq=8 ttl=127 time=1.27 ms
12  1468 bytes from 192.168.2.2: icmp_seq=9 ttl=127 time=1.25 ms
13  1468 bytes from 192.168.2.2: icmp_seq=10 ttl=127 time=1.27 ms
14
15  --- 192.168.2.2 ping statistics ---
16  10 packets transmitted, 10 received, +2 duplicates, 0% packet loss, time 18006ms
17  rtt min/avg/max/mdev = 1.254/1.303/1.427/0.067 ms
```

위에서 NESTNET을 이용하여 우리가 원하는 적절한 WAN환경의 network을 구축을 할 수 있다는 것을 확인 하였다.

NISTNET의 정확한 동작상태를 확인하기 위해 적용된 환경변수(drop packet, delay, duplicate등)를 복합적으로 적용하여 자신이 원하는 network을 구성 할 수 있을 것이다. NESTNET은 network application 과 protocol 등의 performance를 측정하기에 아주 매력 있는 도구라 생각된다.