



기술문서

SaaS OverCloud 환경기반 고속 데이터 전송을 위한 DTN 통합 모니터링 환경 구축 개발

저자 : 문정훈, 박종선
{jhmoon, jspark}@kisti.re.kr
2018.09.12

목차

1. 서론	1
2. 실험 관련 내용 사전조사	2
2-1. ScienceDMZ	2
2-2. PerfSONAR	3
3. 네트워크 관리 소프트웨어(perfSONAR) 설치와 원격 서버 관리	5
인터페이스(IPMI), 실시간 모니터링 툴(Grafana) 구성 순서	
3-1. perfSONAR를 구동시키기 전 환경설정	5
3-2. perfSONAR 테스트 구성	7
3-3. Maddash 환경설정	10
3-4. IPMI 환경설정	11
3-5. Grafana 환경설정	12
3-6. Grafana와 PostgreSQL 연동	13
4. 실험 및 분석결과	15
4-1. 실험 데이터 수집 방법	15
4-2. 실험 및 분석 결과	15
5. 결론	19

그림 목차

그림 1. ScienceDMZ의 구조 모식도	2
그림 2. perfSONAR의 구조 모식도	3
그림 3. DTN 장비 간 네트워크 성능 테스트에서 측정한 결과 그래프	15
그림 4. DTN과 Lustre 간 네트워크 성능 테스트에서 측정한 결과 그래프 ...	16
그림 5. Maddash로 표현한 Host들의 실시간 네트워크 상황(OWAMP)	16
그림 6. Maddash로 표현한 Host들의 실시간 네트워크 상황(BWCTL)	17
그림 7. 오픈 소스 Grafana를 사용하여 실시간 네트워크 상황을 표현한 그림	17
그림 8. Grafana의 Metric으로 생성하기 위한 PostgreSQL의 SQL 구문	17
그림 9. ScienceDMZ 기반 글로벌 DTN 성능 측정을 위한 Maddash 테이블 ..	18

표 목차

표 1. perfSONAR에서 설정한 테스트 세부 내용	15
--------------------------------------	----

1. 서론

최근 빅 데이터에 대한 많은 연구들이 활발히 이루어지고 있다. 본 보고서에서는 비약적으로 발전하고 있는 빅데이터 분야의 고속 전송을 위한 기술 개발로서 데이터 집약형 과학분야의 빅데이터는 관측, 실험, 분석 장비의 발전으로 데이터의 양이 폭증함에 따라 현재의 네트워크 환경과 전송 기술로서는 여러 가지 제약들로 인해 주어진 대역폭을 최대한 활용하는 고속 전송 환경을 갖추기 어려운 상황이므로 이에 대한 환경을 구축하기 위한 절차 방안을 설명하고자 한다. 따라서 이러한 전송 환경의 개선과 주어진 대역폭 대비 최적의 전송을 위해 미국의 ESnet에서 제안한 ScienceDMZ 기반의 전송 환경의 구축을 통해 과학 빅데이터의 초고속 전송, 공유 및 저장을 하며 [1] 과학 빅데이터의 장거리 전송을 위해서는 고대역 네트워크 환경 뿐만 아니라 네트워크 성능을 급격히 저하시키는 Packet-Loss의 발생을 최소화 함으로써 빅데이터의 초고속 전송이 가능하게 되며, 빅데이터의 전송을 위한 전송 가속 시스템인 DTN의 고속화/최적화 튜닝을 통해 이루어진다. 따라서 전송 가속 시스템인 DTN과 고대역 네트워크에 대한 통합 성능 모니터링 체계는 과학 빅데이터의 효과적 전송, 저장, 공유를 위해 매우 중요하며 이러한 시스템과 네트워크에 대한 통합형 성능 모니터링 기술은 향후 대륙간 100G 네트워크 기반의 초고속 성능 DTN의 구축 개발로 Peta-Scale DTN 구축과 전송을 가능케한다.

따라서 본 보고서에서는 이러한 과학 빅데이터 기반의 트래픽 네트워크 성능 모니터링을 위해 미국의 ESnet에서 개발한 PerfSONAR를 통해 모니터링 하며, 시스템을 IPMI를 통해 모니터링한다. 또한, PerfSONAR와 IPMI를 통해 실시간 모니터링한 결과들은 Grafana를 통해 가시화시킴으로써 효과적인 통합 성능 모니터링 환경을 구현한다. 특히 PerfSONAR는 Active/Passive Measurement를 동시에 수행할 수 있으며, 패킷 카운터를 통해 현재 측정 구간의 네트워크 저하 요소인 Packet-Loss의 발생에 따른 적절한 대응이 가능하며, 또한 IPMI를 통해 DTN 노드의 CPU, 메모리, 하드디스크, 네트워크 사용 등을 모니터링 함으로써 효과적인 노드 관리가 가능하다.

2. 실험 관련 내용 사전조사

본 실험은 앞서 언급한 바와 같이 네트워크 관리 툴킷인 PerfSONAR를 사용하여 DTN 노드들로 구성된 네트워크에서 성능을 측정하였으며 측정된 결과들을 기반으로 Maddash와 Grafana를 통해 실시간 네트워크 성능을 가시화하였다.

2-1. ScienceDMZ

최근 빅데이터 분야와 관련된 기술이 급증함에 따라 이러한 빅데이터를 효율적으로 관리하고 분석하며 저장하는 기술이 각광받고 있다. 실생활에서 사용되고 있는 네트워크 환경에선 다양한 목적의 트래픽이 존재한다. 크게 사무용, 업무용 목적의 트래픽인 Business Traffic과 연구용, 교육용 목적의 Research Traffic으로 구분할 수 있는데 이 두 트래픽을 한 네트워크 망 내에서 함께 사용하고 있기 때문에 이로 인하여 트래픽 간에 충돌이 일어날 가능성 또한 매우 높으며 네트워크의 연결거리와 지연시간 차이에 따라 물리적인 전송 성능 또한 저하될 우려가 높다. 이러한 빅데이터 기반의 네트워크 단점을 보완하고자 출현한 매커니즘이 바로 ScienceDMZ 기술이다. ScienceDMZ는 대용량 과학 데이터 전송 구조 기술로써 연구/교육 목적의 데이터를 고속으로 전송하기 위해 기존 네트워크 환경에서 트래픽의 특성 별로 망을 분리하는 구조를 갖고 있다. 따라서 위 언급한 네트워크 성능 저하 문제를 해결할 수 있으며 보다 경량화된 네트워크 보안 기술을 적용하고 있다는 장점을 가지고 있다.

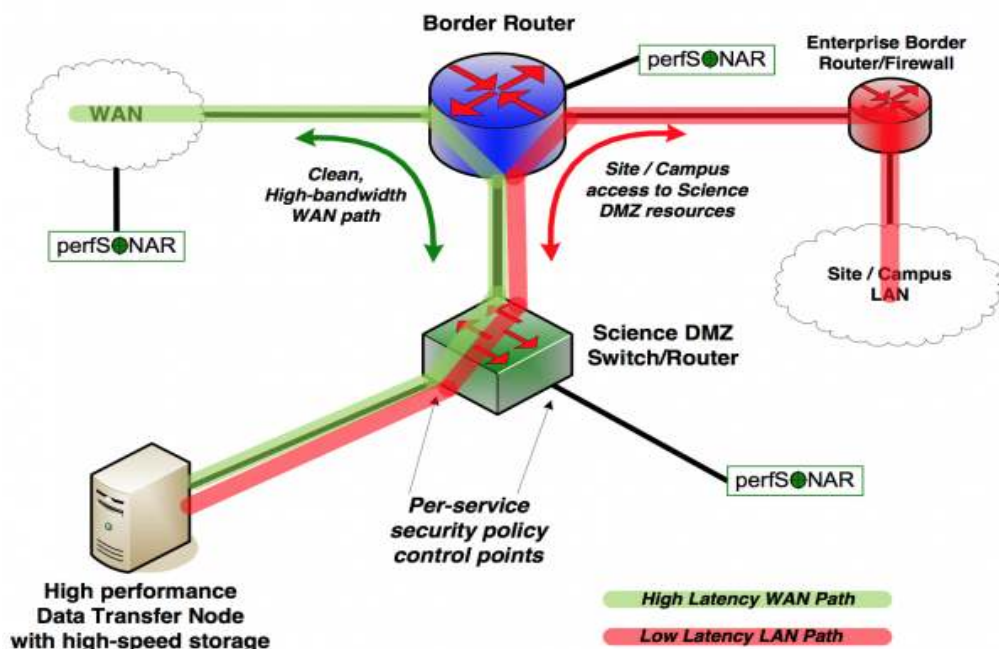


그림 1. ScienceDMZ의 구조 모식도

그림 1은 ScienceDMZ의 구조 모식도로써, 중앙에 있는 Border Router를 기준으로 왼쪽으로 업무용 트래픽, 오른쪽에 연구용 트래픽으로 각각 망이 분리되는 것을 확인할 수 있다. 따라서 ScienceDMZ에선 과학 빅데이터를 고속으로 전송하기 위한 기술이기 때문에 네트워크 망의 Throughput과 Packet Loss가 매우 중요한 변수로 작용된다.

2-2. PerfSONAR

perfSONAR(PERFORMANCE Service Oriented Network monitoring Architecture)는 중단간 네트워크 경로의 적용 범위를 제공하고 네트워크 성능을 측정하는데 도움이 되도록 설계된 Network Management Toolkit이다. 전 세계적으로 배포된 1000개 이상의 perfSONAR 인스턴스가 존재하며, 그 중 대다수는 네트워크의 성능을 공개적으로 테스트할 수 있다. 이러한 글로벌 인프라는 네트워크 내에서 문제가 발생했을 때 이를 식별하고 에러 상황을 격리하여 조치할 수 있으며, 네트워크 리소스를 활용할 때 네트워크 성능을 향상시키는데 도움을 줄 수 있다. perfSONAR는 특히 measurement scheduling, 데이터 저장, 데이터 검색 및 MaDDash를 사용하여 시각화를 할 수 있도록 다양한 인터페이스를 제공한다.

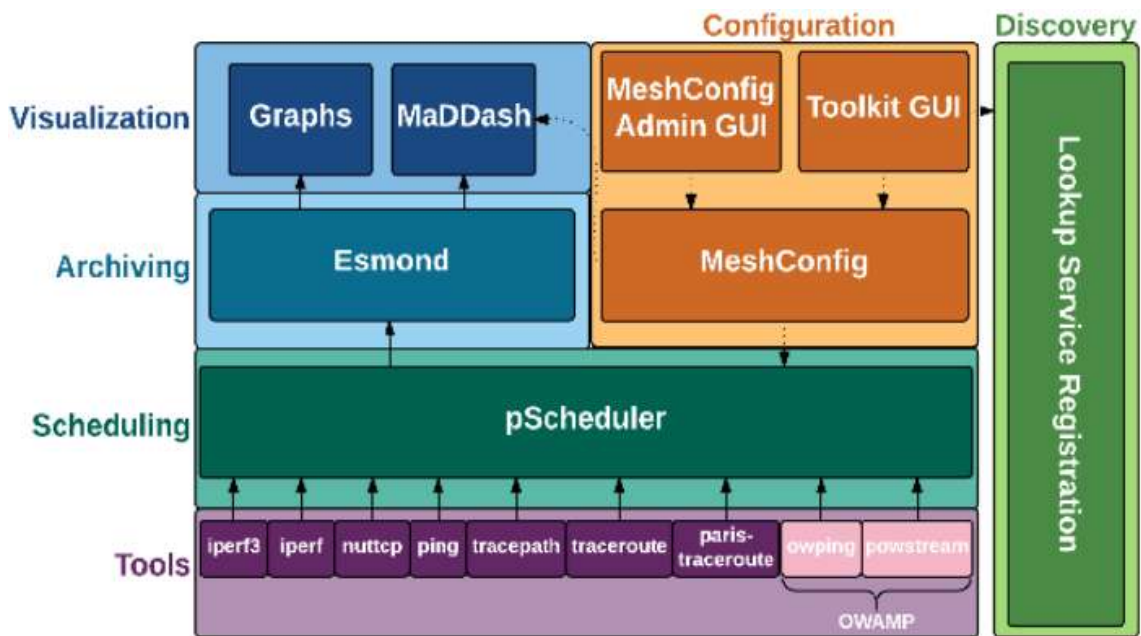


그림 2. perfSONAR의 구조 모식도

또한 perfSONAR의 네트워크 성능 측정 방식은 Active Measurement와 Passive Measurement로 각각 나눌 수 있다. Active Measurement는 테스트 패킷같이 패킷을 발생시켜서 망의 성능을 관찰하는 방식이다. 따라서 네트워크 망에 부가적인 트래픽을 발생시킴으로써 망의 성능을 저하시킬 가능성이 있으며 인터넷 망의 동적인 라우팅 특성 때문에 측정 성능이 실제 망 성능보다 더 우수하게 또는 더 저하되어 나타날 수 있다는 단점이 있다. 그리고 Passive Measurement는 추가적인 패킷 발생없이 망에 흐르는 트래픽을 수집하고 그 트래픽의 특성을 분석하는 방식이다. 이 측정 방법의 용도는 망의 사용량 변화와 패킷 분석을 통한 네트워크 망 제어, 트래픽 사용량 측정 등에 사용될 수 있다. 하지만 Passive Measurement는 대용량 데이터의 발생으로 인해 분석에 많은 자원을 필요로 하고 망 고속화에 대한 대응이 쉽지 않다는 단점이 존재한다. 그림 2는 perfSONAR의 내부 구조를 한눈에 확인할 수 있도록 표현한 모식도이다. 이 구조에 있는 Tool 중에 ping, tracepath, traceroute tool 들이 Active Measurement 방식을 사용하여 네트워크 성능을 측정하며, 네트워크의 Throughput을 측정하는 도구인 iperf, iperf3 그리고 Packet Loss를 측정하는 owamp가 Passive Measurement 측정 방식을 사용하는 대표적인 예이다.

perfSONAR에서 제공하는 소프트웨어 중에 Esmond라는 아카이브 소프트웨어가 존재한다. 이 소프트웨어의 역할은 네트워크에서 측정된 결과 데이터들을 저장하기 위한 목적으로 활용되며 Esmond는 각각 Cassandra 데이터베이스와 PostgreSQL 데이터베이스로 구성되어 있다. Cassandra 데이터베이스는 측정 결과를 저장하는 데이터베이스로 데이터 복제, 장애 극복, 로드 밸런싱, 시간 지연에 따라 노드를 추가/제거/교체하는 기능을 제공한다. PostgreSQL 데이터베이스는 측정 메타 데이터를 저장하는 데이터베이스이며 Esmond에 새로운 정보를 추가하는데 사용되는 사용자의 이름, API 키, 공인 IP 주소 등이 각각 저장되어 있다. 따라서 본 실험에서는 DTN 노드들 간 네트워크 내에서 측정된 성능 데이터들을 기반으로 실시간 모니터링 환경을 구축하기 위해 Cassandra 데이터베이스에 수집된 결과 데이터들을 사용하여 실시간 모니터링 오픈 소스 소프트웨어인 Grafana에 가시화하였다.

3. 네트워크 관리 소프트웨어(perfSONAR) 설치와 원격 서버 관리 인터페이스(IPMI), 실시간 모니터링 툴(Grafana) 구성 순서

3-1. perfSONAR를 구동시키기 전 환경설정

▶ JAVA jdk 7 설치

```
# cd /usr/local
# sudo mkdir tar
# http://www.oracle.com/technetwork/java/javase/downloads/index.html 접속
Java Archive에서 jdk-7u80-linux-x64.tar.gz 파일을 /usr/local/tar 경로로 다운로드
# cd /usr/local/tar
# sudo tar xvfz jdk-7u80-linux-x64.tar.gz
# sudo mv jdk1.7.0_80 /usr/local/
# cd /usr/local
# sudo ln -s jdk1.7.0_80 java
# sudo nano /etc/profile
JAVA_HOME=/usr/local/java
CLASSPATH=.:$JAVA_HOME/lib/tools.jar
PATH=$PATH:$JAVA_HOME/bin
export JAVA_HOME CLASSPATH PATH
# sudo source /etc/profile
# echo $JAVA_HOME
# update-alternatives --config java
```



```
[kisti@203 local]$ sudo update-alternatives --config java
2 개의 프로그램이 'java'를 제공합니다.
선택   명령
-----
1      java-1.7.0-openjdk.x86_64 (/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.191-2.6.15.4.el7_5.x86_64/jre/bin/java)
** 2   java-1.8.0-openjdk.x86_64 (/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.181-3.b13.el7_5.x86_64/jre/bin/java)
현재 선택(*)을 유지하려면 엔터키를 누르고, 아니면 선택 번호를 입력하십시오:1
```

여기서 1번을 선택하여 java version을 1.7.0으로 변경

```
# javac -version
```

```
# java -version
```

javac는 1.7.0_80, java는 openjdk-version-1.7.0_80이 나와야 정상 구성

▶ Apache Tomcat 6 설치

```
# cd /usr/local/tar
# sudo wget http://archive.apache.org/dist/tomcat/tomcat-6/v6.0.53/bin/apache-tomcat-6.0.53.tar.gz
# sudo tar xvfz apache-tomcat-6.0.53.tar.gz
# sudo mv apache-tomcat-6.0.53 /usr/local
# cd /usr/local
# ln -s apache-tomcat-6.0.53 tomcat
# sudo nano /etc/profile
  JAVA_HOME=/usr/local/java
  JRE_HOME=/usr/local/java
  CATALINA_HOME=/usr/local/tomcat
  CLASSPATH=.:$JAVA_HOME/lib/tools.jar:$CATALINA_HOME/lib/jsp-api.jar:$CATALINA_H
OME/lib/servlet-api.jar
  PATH=$PATH:$JAVA_HOME/bin:$CATALINA_HOME/bin
  export JAVA_HOME CLASSPATH PATH CATALINA_HOME JRE_HOME
# sudo source /etc/profile
# echo $CATALINA_HOME
# /usr/local/tomcat/bin/startup.sh
# netstat -an | grep 8080
# /usr/local/tomcat/bin/shutdown.sh
# firewall-cmd --permanent --zone=public --add-port=8080/tcp
# firewall-cmd --reload
# firewall-cmd --permanent --list-all
# Web Browser에서 http://localhost:8080 으로 접속
```

▶ Apache Tomcat 6 정상 구동 시 화면



3-2. perfSONAR 테스트 구성

▶ perfSONAR 관련 패키지 설치

```
# sudo yum install epel-release
# sudo yum install http://software.internet2.edu/rpms/el7/x86_64/main/RPMS/perfSONAR-repo-0.8-1.noarch.rpm
※ Error : One of the configured repositories failed (unknown)
sudo yum clean all
sudo rm -rf /var/cache/yum
sudo yum clean metadata
dhclient
sudo nano /etc/resolv.conf
    nameserver 168.126.63.1
    nameserver 8.8.8.8
    nameserver 8.8.4.4

# sudo yum install perfsonar-tools -y
※ Error : python-devel conflicts with python... : Could not install OS dependencies
package-cleanup --dupes
package-cleanup --cleandupes
package-cleanup --problems
yum update -y

# sudo yum install perfsonar-testpoint -y
# sudo yum install perfsonar-core -y
# sudo yum install perfsonar-centralmanagement -y
# sudo yum install perfsonar-toolkit -y
# sudo /usr/lib/perfsonar/scripts/install-optional-packages.py
# sudo /usr/lib/perfsonar/scripts/configure_firewall install
# sudo systemctl enable yum-cron
# sudo systemctl start yum-cron
# service 상태 확인
sudo systemctl status pscheduler-scheduler
sudo systemctl status pscheduler-runner
sudo systemctl status pscheduler-archiver
sudo systemctl status pscheduler-ticker
```

```
sudo systemctl status owamp-server
sudo systemctl status bwctl-server
sudo systemctl status perfsonar-lsregistrationdaemon
```

※ bwctl-server 상태를 확인하는 과정에서 오류난 경우

perfSONAR 버전이 4.0 이상이면 bwctl-server를 지원하지 않고 Bandwidth 측정하는 도구인 iperf와 iperf3를 pscheduler에 포함시킴. 따라서 4.0 버전 이상일 경우 bwctl-server의 상태 확인은 하지 않아도 됨. 아래 명령어는 bwctl이 동작하지 않는 경우 실행.

```
sudo yum install bwctl bwctl-server bwctl-client -y
```

```
sudo wget http://software.internet2.edu/sources/bwctl/bwctl-1.5.1-20.tar.gz
```

```
sudo tar xvfz bwctl-1.5.1-20.tar.gz
```

```
cd bwctl-1.5.1
```

```
sudo ./configure --prefix=/ami
```

```
sudo make
```

```
sudo make install
```

```
sudo systemctl status bwctl-server.service
```

```
# sudo nano /etc/selinux/config
```

SELINUX=disabled 로 변경

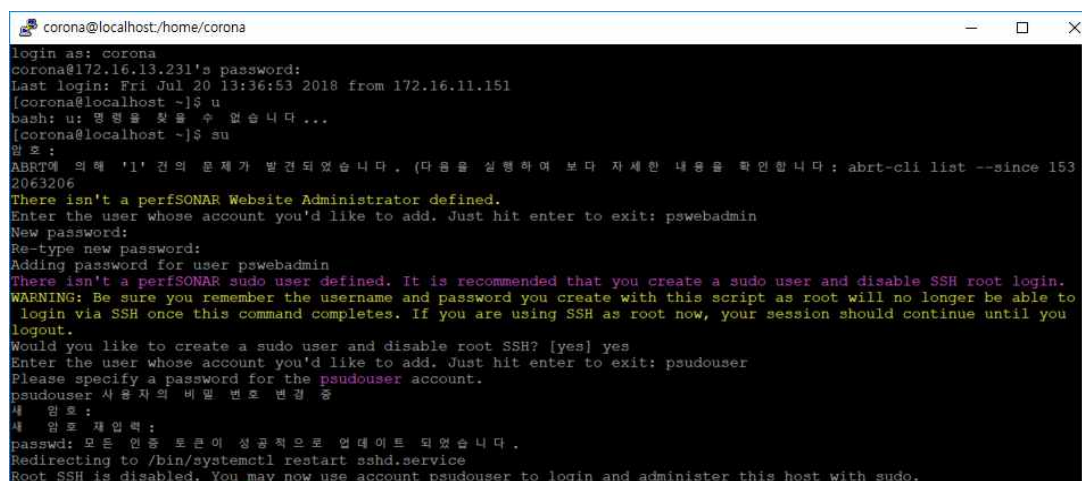
```
# sudo systemctl enable NetworkManager
```

```
# sudo systemctl start NetworkManager
```

```
# /sbin/chkconfig NetworkManager on
```

```
# su 관리자 계정으로 접속
```

▶ SSH연결 프로그램(Putty)으로 관리자 계정 접속한 화면



```
corona@localhost/home/corona
login as: corona
corona@172.16.13.231's password:
Last login: Fri Jul 20 13:36:53 2018 from 172.16.11.151
[corona@localhost ~]$ u
bash: u: 명령을 찾을 수 없습니다...
[corona@localhost ~]$ su
암호:
ABRT에 의해 '1' 권의 문제가 발견되었습니다. (다음을 실행하여 보다 자세한 내용을 확인합니다: abrt-cli list --since 1532063206)
There isn't a perfSONAR Website Administrator defined.
Enter the user whose account you'd like to add. Just hit enter to exit: pswebadmin
New password:
Re-type new password:
Adding password for user pswebadmin
There isn't a perfSONAR sudo user defined. It is recommended that you create a sudo user and disable root SSH.
WARNING: Be sure you remember the username and password you create with this script as root will no longer be able to login via SSH once this command completes. If you are using SSH as root now, your session should continue until you logout.
Would you like to create a sudo user and disable root SSH? [yes] yes
Enter the user whose account you'd like to add. Just hit enter to exit: psdouser
Please specify a password for the psdouser account.
psdouser 사용자의 비밀번호 변경 중
새 암호:
새 암호 재입력:
passwd: 모든 인증 토론이 성공적으로 업데이트되었습니다.
Redirecting to /bin/systemctl restart sshd.service
Root SSH is disabled. You may now use account psdouser to login and administer this host with sudo.
```

관리자 ID는 pswebadmin, 사용자 ID는 psdouser로 입력함.

▶ Web Browser에서 http://localhost/toolkit 으로 접속한 화면

The screenshot shows the perfSONAR Toolkit interface for IP 203.253.235.139. It includes a top navigation bar with 'View public dashboard', 'Configuration', and 'Help'. The main content area is divided into several sections:

- Organization:** OXLab, Address: Seoul, Korea, Administrator: Yonghak Kim (ykim@ox.ac.kr)
- Services:** A table listing services like 'twiki', 'diamond', 'registration', 'meshonfig-agent', 'eswamp', and 'gscheduler' with their status (Running), version, and ports.
- Test Results:** A section for viewing test results, including a search bar and a table of test results.
- Health:** A section showing system health metrics like CPU usage, disk usage, and memory usage.
- Host Information:** A section showing host details like interfaces, primary interface, and globally registered status.

※ Login 시 Error : SEC_ERROR_UNKNOWN_ISSUER 발생했을 경우

Web Browser (ex. Firefox) 주소 창에서 aboutconfig를 입력한 뒤 security.enterprise-roots.enabled=true로 변경

계속 오류가 발생하면 해당 창에서 Advanced (고급) 버튼을 클릭하고 Exception(예외 사항)에 localhost/toolkit/auth/를 영구적으로 추가할 것. 그 후 Login 창이 발생하면 위에서 설정해준 관리자 ID와 비밀번호를 통해 접속.

▶ perfSONAR UI 화면에서 Scheduler Test 구성 절차

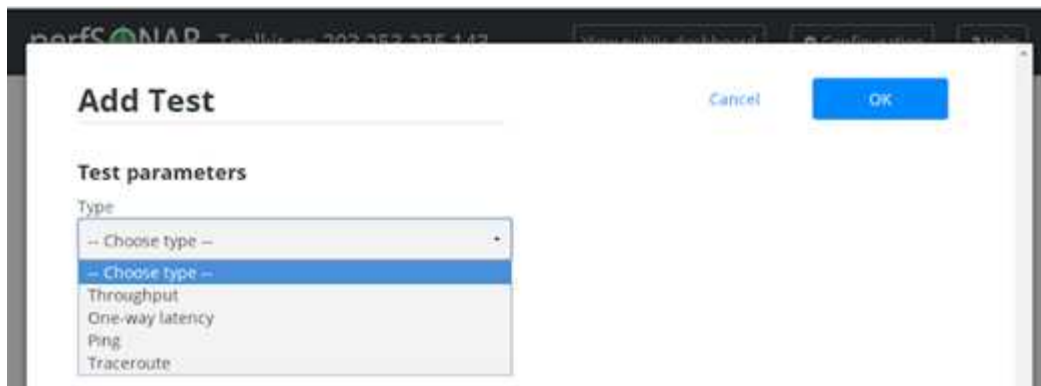
Web Browser에서 http://localhost/toolkit으로 접속

중앙에 Test Results 오른쪽 Configure tests 클릭

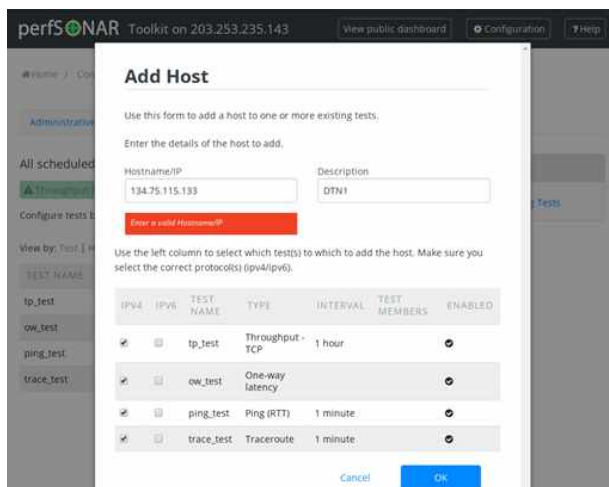
The screenshot shows the 'Configuration / Tests' page in the perfSONAR Toolkit. It includes a top navigation bar with 'Home', 'Configuration', and 'Tests'. The main content area is divided into several sections:

- Administrative information:** A section for administrative information.
- Host:** A section for host configuration.
- Tests:** A section for test configuration.
- All scheduled tests:** A section showing all scheduled tests, including a warning that throughput tests will be running 0% of the time.
- Resources:** A section for resources, including a 'Configuring Tests' button.

4가지 Test 추가 (Throughput, One-way Latency, Ping, Traceroute)



Host 입력 후 Test에 각각 추가



오른쪽 하단에 Save 버튼 클릭

3-3. Maddash 환경설정

▶ Maddash 설치

```
# sudo yum install maddash -y
```

```
# sudo nano /etc/maddash/maddash-server/maddash.yaml
```

Ctrl+W (구문탐색) 으로 myOwampHosts와 myBwctlHosts에 각각 node 주소 입력

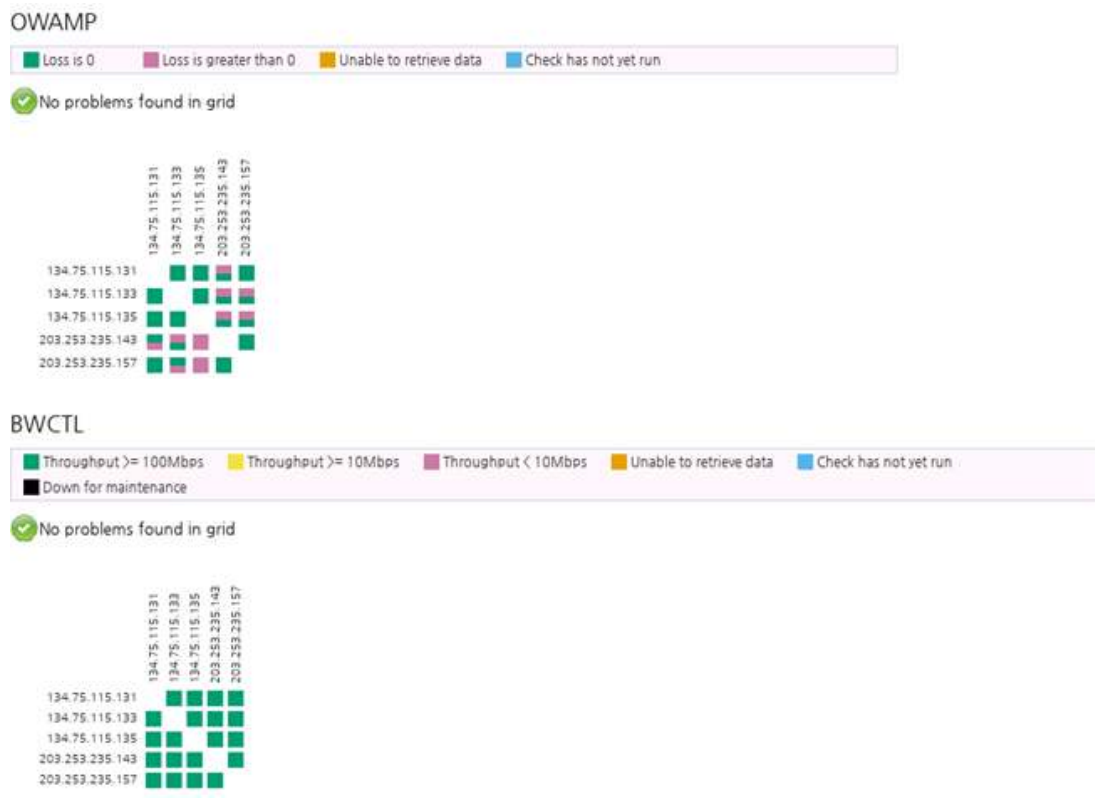
Ctrl+W (구문탐색) 으로 example.mydomain.local을 perfSONAR Toolkit이 설치되어 있는 Host 주소로 변경 (owampLossCheck, owampLossRevCheck, bwctlCheck, bwctlRevCheck의 graphUrl)

※ 만약 **maddash.yaml** 파일이 비어있는 경우

<http://github.com/esnet/maddash/blob/master/maddash-server/etc/maddash.yaml>
에서 기본적으로 구성되어 있는 maddash.yaml 파일을 다운로드 받는 것을 추천함.

```
# htpasswd -c /etc/maddash/maddash-webui/admin-users myadmin  
Root ID : myadmin
```

▶ <http://localhost/maddash-webui> 접속 시 화면



3-4. IPMI 환경설정

▶ IPMI 설치

```
# sudo yum install ipmitool OpenIPMI OpenIPMI-tools -y  
# lsmod | grep ipmi  
ipmi_si / ipmi_devintf / ipmi_msghandler 동작 확인  
# sudo systemctl start ipmi  
# sudo systemctl enable ipmi
```

```
# sudo /sbin/chkconfig ipmi on
```

▶ ipmi 명령어를 통해 원격으로 장치를 재부팅하는 화면

```
[root@l34 kisti]# ipmitool -I open chassis status
System Power           : on
Power Overload         : false
Power Interlock        : inactive
Main Power Fault       : false
Power Control Fault    : false
Power Restore Policy   : previous
Last Power Event       :
Chassis Intrusion      : inactive
Front-Panel Lockout    : inactive
Drive Fault            : false
Cooling/Fan Fault      : false
Sleep Button Disable   : not allowed
Diag Button Disable    : allowed
Reset Button Disable   : not allowed
Power Button Disable   : allowed
Sleep Button Disabled  : false
Diag Button Disabled   : true
Reset Button Disabled  : false
Power Button Disabled  : false
[root@l34 kisti]# ipmitool -I open power reset
Chassis Power Control: Reset
[root@l34 kisti]#

Session stopped
- Press <return> to exit tab
- Press R to restart session
- Press S to save terminal output to file

Network error: Software caused connection abort
```

3-5. Grafana 환경설정

▶ Grafana 패키지 설치

```
# sudo wget https://s3-us-west-2.amazonaws.com/grafana-releases/release/grafana-5.1.4-1.x86_64.rpm
# sudo yum install initscripts fontconfig
# sudo rpm -Uvh grafana-5.1.4-1.x86_64.rpm
# sudo nano /etc/yum.repos.d/grafana.repo

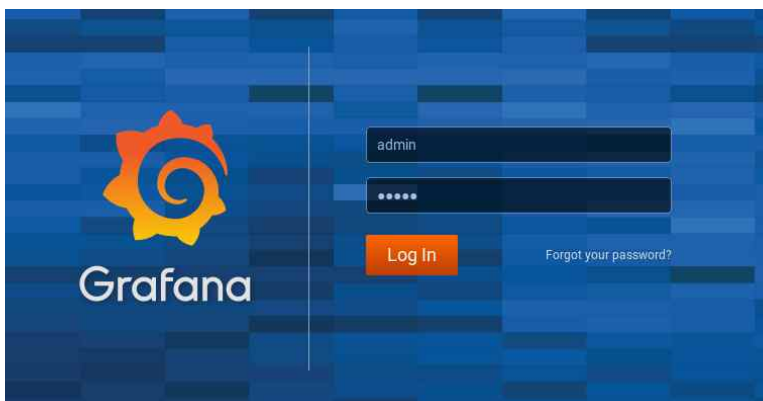
[grafana]
name=grafana
baseurl=https://packagecloud.io/grafana/stable/el/7/$basearch
repo_gpgcheck=1
enabled=1
gpgcheck=1
gpgkey=https://packagecloud.io/gpg.key https://grafanarel.s3.amazonaws.com/RPM-GPG-KEY-grafana
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt

# sudo yum install grafana -y
```



```
# sudo systemctl start grafana-server
# sudo /sbin/chkconfig --add grafana-server
# sudo systemctl daemon-reload
# sudo systemctl start grafana-server
# sudo systemctl status grafana-server
# sudo systemctl enable grafana-server.service
```

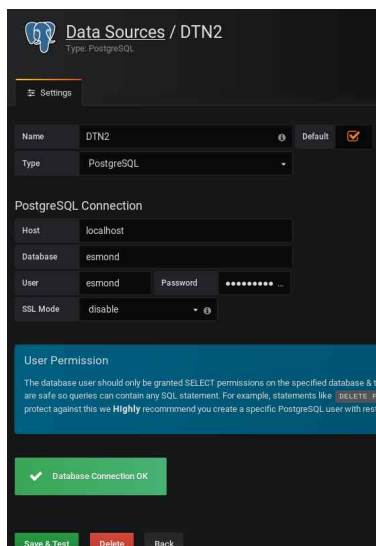
▶ Web Browser를 통해 <https://localhost:3000> 으로 접속



최초 ID : admin / PW : admin

3-6. Grafana와 PostgreSQL 연동

▶ Data Source 설정 화면



Name : 사용자 임의로 설정 (DTN2)

Type : PostgreSQL

Host : PostgreSQL Database가 있는 주소로 설정
(내부에 구축했으므로 Localhost)

Database Name : esmond

User ID : esmond

※ Password는 terminal을 통해 특정 파일(/etc/esmond/esmond.conf)을 열어서 확인

```
GNU nano 2.3.1 File: esmond.conf
[main]
sql_db_engine = django.db.backends.postgresql_psycopg2
sql_db_name = esmond
sql_db_user = esmond
sql_db_password = 2vCnwxmP9wu41iE6kevJYjTnWl54nvqd
tsdb_root = %(ESMOND_ROOT)s/tsdb-data
tsdb_chunk_prefixes = %(ESMOND_ROOT)s/tsdb-data
```

sql_db_password 항목에 있는 내용으로 입력

SSL Mode는 지원하지 않으므로 disable로 설정하고 하단에 Save & Test 버튼 클릭

※ /etc/esmond/esmond.conf 파일을 수정했을 경우

PostgreSQL Database에 영향을 받아 HTTP/1.1 500 Internal Server Error가 발생함.
특히, pscheduler를 통해 측정된 throughput, Latency, RTT 값들이 저장되는 PostgreSQL Database 경로를 찾지 못하여 값이 저장되지 않음. 따라서 esmond.conf 파일은 변경하지 않을 것을 추천함.

lsmod | grep ipmi

ipmi_si / ipmi_devintf / ipmi_msghandler 동작 확인

sudo systemctl start ipmi

sudo systemctl enable ipmi

sudo /sbin/chkconfig ipmi on

▶ ipmi 명령어를 통해 원격으로 장치를 재부팅하는 화면

```
[root@134 kisti]# ipmitool -I open chassis status
System Power           : on
Power Overload         : false
Power Interlock        : inactive
Main Power Fault       : false
Power Control Fault    : false
Power Restore Policy   : previous
Last Power Event       :
Chassis Intrusion      : inactive
Front-Panel Lockout    : inactive
Drive Fault            : false
Cooling/Fan Fault      : false
Sleep Button Disable   : not allowed
Diag Button Disable    : allowed
Reset Button Disable   : not allowed
Power Button Disable   : allowed
Sleep Button Disabled  : false
Diag Button Disabled   : true
Reset Button Disabled  : false
Power Button Disabled  : false
[root@134 kisti]# ipmitool -I open power reset
Chassis Power Control: Reset
[root@134 kisti]#

Session stopped
- Press <return> to exit tab
- Press R to restart session
- Press S to save terminal output to file

Network error: Software caused connection abort
```

4. 실험 및 분석결과

4-1. 실험 데이터 수집 방법

표 1. perfSONAR에서 설정한 테스트 세부 내용

Test Types	Throughput	Ping	Traceroute
Time Between tests	6 Hours	5 Minutes	10 Minutes
Test Durations	20 sec	.	.
Packets per tests	.	10	.
Packet Size (Bytes)	.	.	40

실험 데이터를 수집하기 위해 다섯 대의 서버에 네트워크 성능(throughput) 측정 프로그램 iperf3를 이용하여 성능 데이터를 수집한다. iperf3를 이용한 성능(throughput) 측정 간격은 1초로 총 시간은 60초간 10번의 명령어를 수행하여 데이터를 텍스트 형식으로 추출한 후 저장한다. 이와 같은 방법으로 스트림의 수를 1 부터 6 까지 변화를 주어 데이터를 구성하였다. 표 1은 perfSONAR에서 네트워크 성능을 측정하기 위해 구성한 테스트들의 세부 내용을 보여주고 있다. 이와 같이 구성된 데이터를 바탕으로 분석을 수행한다.

4-2. 실험 및 분석결과

본 실험은 모두 Cent OS 7 기반으로 된 Dell PowerEdge R730 모델 3대(이하 DTN)와 Dell PowerEdge R720 모델 2대(이하 Lustre)를 사용하여 수행하였다. DTN 장비들은 Mellanox Technologies Ethernet 40 Gb/s, Lustre 장비들은 Ethernet 1 Gb/s 의 유선 네트워크망으로 구축되었다. 따라서 본 실험은 총 세 가지 사례로 네트워크 성능 변화를 관찰할 수 있었다. 아래 그림은 DTN 간에 네트워크 성능을 관찰하였을 때 수집된 결과이다.

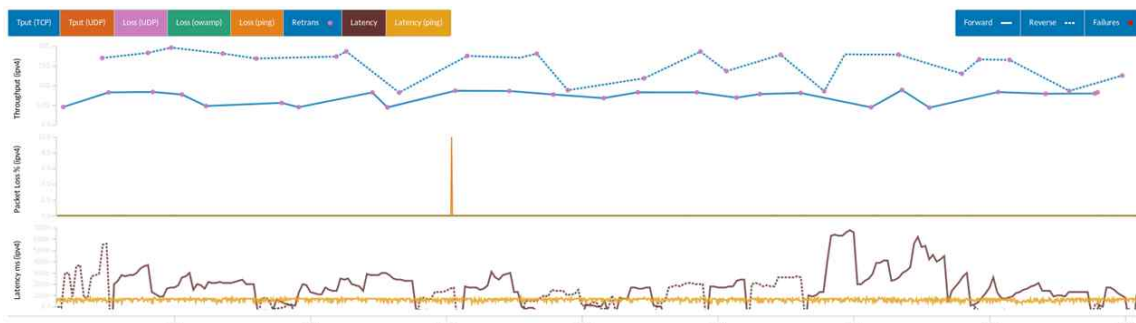


그림 3. DTN 장비 간 네트워크 성능 테스트에서 측정된 결과 그래프

파란색 실선으로 표현된 값이 특정 장비에서 송신했을 때의 처리량이며 파란색 점선으로 표현된 값은 DTN에서 수신했을 때의 처리량 임을 그림 3을 통해서 보여주고 있다. 따라서 네트워크 처리량(Throughput)은 평균 13Gb/s 로 나타나고 있으며 송·수신간에 네트워크 처리량이 다소 차이나는 것을 확인할 수 있다. 이는 Active Measurement 측정 방식과 Passive Measurement 측정 방식을 혼용해서 발생한 문제점이다. 앞서 언급한 바에 의하면 본 실험 장비들에 네트워크 성능 테스트를 구성할 때, Throughput 테스트만 구성한 것이 아니라 ping, traceroute 테스트도 같이 구성하였기에 송신하며 얻어진 처리량은 다소 낮게 측정된 것으로 판단된다.

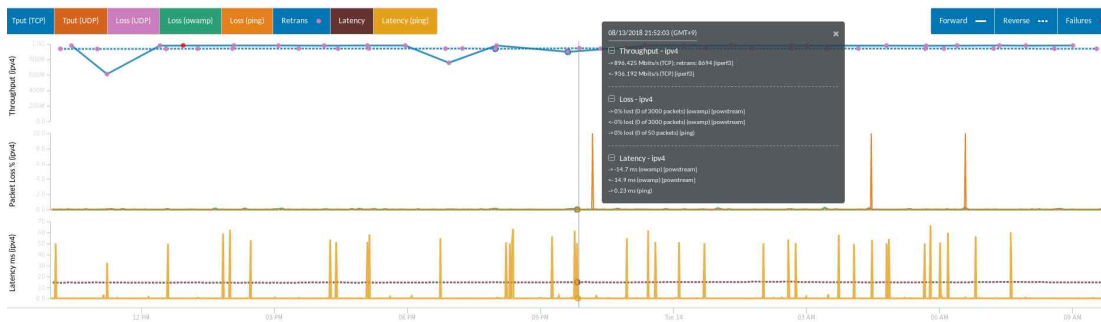


그림 4. DTN과 Lustre 간 네트워크 성능 테스트에서 측정된 결과 그래프

위 그래프 그림 4는 DTN에서 Lustre로 네트워크 성능을 측정된 그래프다. 전송 속도가 DTN에 비해 상대적으로 낮은 Lustre의 전송 속도에 맞춰서 처리량이 평균 980Mb/s 로 측정되는 것을 확인할 수 있다.

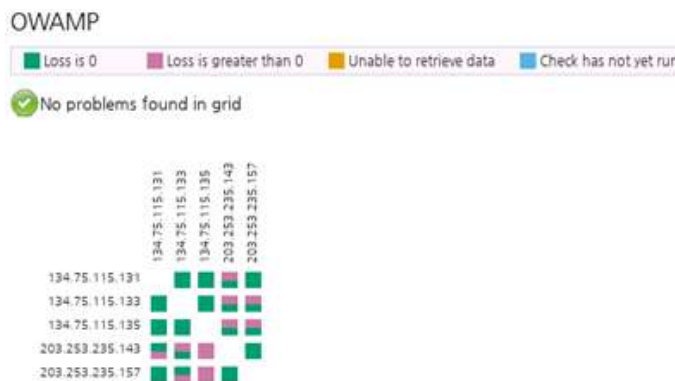


그림 5. Maddash로 표현한 Host들의 실시간 네트워크 상황 (OWAMP)

그림 5는 각 장비들 간에 형성된 네트워크에서 실시간 상황을 단방향지연 측정 도구 (OWAMP)를 사용하여 Maddash로 표현한 그림이다. Packet Loss는 평균적으로 0%

에 가깝게 측정되고 있다.



그림 6. Maddash로 표현한 Host들의 실시간 네트워크 상황 (BWCTL)

그림 6는 각 장비들 간에 형성된 네트워크에서 실시간 상황을 처리량 측정 도구 iperf3와 iperf를 사용하여 Maddash로 표현한 그림이다. 앞서 언급하였듯, DTN 간 네트워크 통신에서 측정된 처리량은 평균 13Gb/s로 측정되며 DTN과 Lustre, Lustre 간 네트워크 통신에서 측정된 처리량은 평균 980Mb/s로 측정되고 있다.

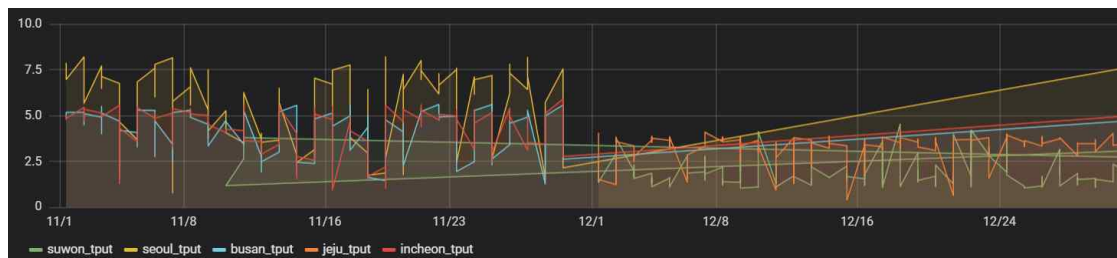


그림 7. 오픈 소스 Grafana를 사용하여 실시간 네트워크 상황을 표현한 그림

```
SELECT
  $__time(times),
  tput_data as Busan_tput
FROM
  public.tput_data
WHERE
  $__timeFilter(times)
  AND
  host_ad='ps.daeg.kreonet.net'
  AND
  source_ad='ps.busn.kreonet.net'
```

그림 8. Grafana의 Metric으로 생성하기 위한 PostgreSQL의 SQL 구문

Maddash를 사용해서 실시간 네트워크 상황을 도표로 표현함과 동시에 오픈 소스인 Grafana를 사용하여 그래프로 표현할 수 있는 사례로 그림 7을 언급하였다. 그림 7와 같이 표현하기 위해선 PostgreSQL 데이터베이스에서 실제 사용되는 SQL 구문을 통해 metrics들을 각각 생성해야 한다. 그림 8은 그림 7에서 그래프로 표현되었던 metrics들 중 대표적인 SQL구문의 예를 보여주고 있다. 그림 7는 KREONET을 기준으로 국내 주요 도시별로 존재하는 KREONET 지역망 센터들 간 연결되어 있는 네트워크 망에서 수집된 데이터들을 기반으로 앞서 설명한 그림 8과 같이 구성된 metrics들을 하나의 Grafana 그래프에 표현한 실시간 네트워크 성능 그래프이다. 그래프가 다소 각진 면이 있지만 이는 문제되는 사항이 아니며 사용자가 원하는 대로 그래프의 metrics들을 부드럽게 표현할 수 있으며, 새 metric을 추가할 수 있다. 단위는 Gb/s로 평균 6~8 Gb/s의 네트워크 처리량이 측정되는 것을 확인할 수 있었다.



그림 9. ScienceDMZ 기반 글로벌 DTN 성능 측정을 위한 Maddash 테이블

이러한 통합형 모니터링 환경의 구축은 이미 전 세계적으로 구축개발이 활발히 진행되고 있는 ScienceDMZ 환경 기반에서 DTN을 통한 빅데이터의 초고속전송 환경에 적용이 가능하며, 그림 9에서 보는 것과 같이 수십 개 이상의 DTN 노드들의 연동에 대한 성능을 모니터링하고 성능을 보장하는 수준까지의 구축 개발이 필요하다. 특히 100G 기반의 전송 환경과 이를 기반으로 Peta-Scale DTN 전송 환경에 대한 시스템과 네트워크 환경의 통합형 성능 모니터링은 전 세계적인 ScienceDMZ의 확장과 성능의 고도화에 따라 매우 중요한 모니터링 이슈이다. 따라서 본 기술문서를 통해 통합형 성능 모니터링을 위한 기술개발의 과정과 적용을 위한 기술을 구현하였다.

5. 결론

본 보고서에서는 최근 급증하는 과학 빅데이터의 초고속 전송을 위한 아키텍처인 ScienceDMZ의 핵심 기능인 DTN (Data Transfer Node)의 전송성능 및 고대역 네트워크 환경의 효과적인 성능 모니터링을 위해 PerfSONAR와 IPMI를 통한 모니터링 결과를 오픈소스인 Grafana를 통한 가시화를 구현하였다. 네트워크 관리 툴킷인 perfSONAR의 환경을 구축하는 방법부터 Maddash와 Grafana의 기본환경을 구축하는 방법을 상세히 설명하였고, perfSONAR 내부에서 DTN과 Lustre File System과의 실제 네트워크 성능 테스트를 통해 얻은 데이터들을 기반으로 그 성능을 분석하였다. 그 후 통합형 모니터링 구축을 위한, 삽입 모듈을 개발하였고, 향후 대륙간 100G 네트워크 환경에서의 DTN을 통한 Peta-Scale 전송 환경에 대한 통합형 모니터링과 세계 각국의 HPC 기반의 ScienceDMZ들 간의 다양한 전송 환경에 대한 통합형 성능 모니터링에 대한 연구 개발이 필요하다.

Reference

- [1] E.Dart et al.. “The ScienceDMZ: A Network Design Pattern for Data-Intensive Science” . Supercomputing Conference 2013. Nov. 17-21. 2013. Denver. CO. USA.
- [2] J. H. Jeong, S. Y. Lee, Y. J. Kim “Methodology and Systems for Internet Traffic Measurement” , Electronics and telecommunications trends, Vol 16. No. 5. p33-35. Oct, 2001
- [3] J. S. Park, S. H. Yoon, S. S. Yoon “Flow-Based Internet Traffic Measurement Technologies” , Electronics and telecommunications trends, Vol 19. No. 6. p93-104. Dec. 2004.
- [4] Multiple tools of perfSONAR, http://docs.perfsonar.net/info_about.html
- [5] Managing Measurement Archive Data, http://docs.perfsonar.net/multi_ma_backups.html
- [6] How to see a Cassandra Database Keyspace, <http://coronasdk.tistory.com/851>
- [7] List available as Grafana Data Source, <http://docs.grafana.org/features/datasources/>
- [8] Cassandra takes advantage of Big Table and Dynamo, http://www.kdata.or.kr/info/info_04_view.html?field=&keyword=&type=techreport&page=38&dbnum=176031&mode=detail&type=techreport