第 VI 部

ネットワーク管理とセキュリティ

第 6 部
ネットワーク管理とセキュリティ

第 1 章 Introduction

The second generation gigabit network JGNII has been launched in April 2004 as a project of the NICT. After a successful experiment with data collection and servicing in a multi-user, multiproject environment on the erstwhile JGN network—the next generation information collection, archiving and servicing project has been launched. The goals are

(1) to build on the experience of the network collection and servicing experiment on JGN
(2) to address the problems encountered
(3) to make the information content richer so that it can service the requirements of a diverse group of users and network researchers
(4) to make the system widely available at an early date

第 2 章 Areas of research and investigation

There are some specific challenges related to the data collection activity that will need to be met. New technology and research will be required to address these challenges. In the following we enumerate some of the areas;

1 research will be carried out on providing new types of information that may help us obtain a correct estimation and/or understanding of the network traffic characteristics. Among the new types of information are;
   a. a host of different statistics at the probe e.g. peak rates measured at small (sub-second) intervals, weighted moving averages, etc.
   b. traffic information aggregated based on attributes like network address (e.g. Aguri) and other metrics.
   c. new 'transforms' of network traffic information where the focus is not on the quantity or volume of the network traffic but on the 'quality' of the traffic (e.g. number of distinct addresses, ports, applications, flows etc.).
   d. network latency.

   The new JGNII network is being used as a test bed for the data collection experiment. Specially designed and developed passive traffic monitors (CpMonitor) have been deployed to generate the new forms of information. Details of this work is appeared in chapter 3.
2 research will be carried out on developing new algorithms to detect 'events of interest' from the statistics. Introduction of this contribution is also appeared in chapter 3.
3 research will be carried out on developing reliable data collection frameworks by introducing redundancy into the data collection system and by developing algorithms to merge two potentially incomplete sets of data to obtain more complete set of data. Details of this work is appeared in chapter 4.
4 a new technology will be developed for updating and maintaining so many monitoring agent remotely on the block. Details of this work is appeared in chapter 5.
5 a new storage technology will be used to archive the collected data in a compressed format and in a lossless manner. The technology should be offer an interface that is compatible with the current data storage formats (viz. sequence of ⟨TimeStamp, MO-value⟩ tuples)

so that current applications can operate in a transparent manner.

第 3 章　Event-based Network Traffic Monitoring In A Wide Area Network

Abstract

Network monitoring is necessary to evaluate the performance and to ensure operational stability and efficiency. We, netman WG, have been monitoring traffic statistics for the JGNII network and are studying the results.

In this chapter, we introduce our monitoring and analysis activities. We have focused on two statistics, one is traffic volume, the other is latency. These statistics provide valuable hints about the underlying network's quality of service and throughput.

We also introduce the concept of "event-oriented network management" and discuss some techniques to detect network events using the above statistics.

3.1 Introduction

Network traffic monitoring is an important aspect of network management and security. For example, observations may reveal the effects of events such as a network failure, an operational failure or a security incident, on network traffic. There are several other usages of network traffic monitoring e.g. in QoS estimation, bandwidth planning etc. But, in routine network monitoring, the interest is on events. If there are no events of interest, the network manager will probably not want to "look" at the traffic. The traffic data in such cases is destined for archiving. From there it will probably be backed-up on off-line media or discarded.

Present monitoring systems do not have a mechanism of detecting events of interest. So it appears that the operator will either look at all the traffic to detect events of interest or will not look at the traffic at all. In our work we attempt to mechanically detect events of interest and draw the operator's attention to these events. We use data from a wide area network to examine the utility and effectiveness of the approach.

The process of mechanical event detection heavily depends on the availability and the accuracy of the data. But in a standard monitoring environment there is little guarantee for these two factors. To raise the availability and accuracy of the data we propose the deployment of multiple data collectors at geographically and network topologically separated points. We have carried out experiments on a wide-area network, and have examined how the quality of the data can be raised i.e. how the availability and accuracy of the data can be increased using the collection redundancy.

In section 3.2 we introduce our monitoring environment. In section 3.3 we examine the issues involved in raising the data availability and accuracy using data from multiple pollers. In section 3.4 we discuss the methods of analyzing the data to detect events. In section 3.7 we describe our ideas of "event-oriented network management".

3.2 Environment

For our work we set up a monitoring environment over the large-scale very high-speed network the Japan Gigabit Network II (JGNII)[150].

JGNII is an open test-bed network environment for research and development and provides nationwide IPv6 network and optical wavelength networks in Japan.

We are executing a project on network traffic monitoring JGNII network. Our aim is to provide network users with network traffic information. We have deployed passive probes which comprise of some tapping equipment. The probes are placed at various sites in Miyagi, Tokyo, Gifu, Kyoto, Hirosihma and Saga. These probes watch the network traffic and generate statistics. The network statistics are collected by data pollers placed in Sendai and Kyoto using the standard

Table 3.1. Monitoring Environment in JGNII

Items	Number
Sites where probe is placed	9
Placed probes	10
Monitoring points	11
Monitoring links	26
(with VLAN)	(19)
Polling Agents	2

Table 3.2. Measuring statistics

IPv6 packets/traffic volume
ICMPv6 packets/traffic volume
TCP over IPv6 packets/traffic volume
UDP over IPv6 packets/traffic volume
Other protocols packets/traffic volume
SNMP Polling Interval
Elapsed time by Traceroute6

network management protocol SNMP.

Table 3.1 shows our monitoring topology as of 26 July 2004. The polling agents at Sendai and Kyoto poll the passive probes every 60 seconds using SNMP over IPv6. The traffic data is available for viewing at [151].

We show the monitoring traffic statistics in Table 3.2. Here "other protocols" denotes packets which had an IPv6 packet header but the next header field is not ICMPv6, TCP or UDP. We have also collected elapsed time information obtained by executing traceroute6 from the data collectors to the probes.

3.3 Multiple Monitoring

In a standard monitoring environment there is little guarantee about the data availability and accuracy. In the data collection process we use SNMP over UDP over IPv6, to collect traffic statistics from the passive probes. UDP does not guarantee the delivery of packets. So packets may get lost. The applications do several retries in case a response is not received. Yet the case of a response being missed due to packet loss may not be ruled out. There may be also be the case of data loss due to problems at the data collector. The data collector application maybe overloaded, or dead, the data collector host may be overloaded or down. To raise the availability and accuracy of the data we deployed two data collectors at geographically and network topologically separated points.

With data from multiple data collectors we attempt to synthesize a data repository that has data availability and accuracy levels greater than or equal to that of the archives of the individual data collectors. First we select a master archive for the traffic statistics by comparing the data contents of the archives of data collectors. The parameters that are considered in selecting the master archive are the number of successful polls, the mean polling interval and the variance of polling interval. The archives that are not a master archive are auxiliary archives. In the next step, we complement the traffic statistics of the master archive with the missing data points, wherever possible, from the auxiliary archives. Finally, wherever there is a fluctuation in the polling interval, the polling interval is normalized by interpolating the traffic data.

In short, our plan to merge traffic statistics comprises of the following steps:

1. Select a master archive
2. Complement the master archive with data from auxiliary archives
3. Normalize the time stamps

3.4 Network Analysis

To mechanically detect events of interest, we use JGNII's traffic statistics. We focus on two statistics viz., traffic volume and latency. We discuss the properties of these statistics and describe techniques to analyze them.

3.5 Traffic volume

Traffic volume may be considered to be as one of the indicators of network status. For example, lack of traffic may indicate some network event like a network fault. On the other hand an unusually large traffic may indicate that a DoS attack is underway. The following figure shows the

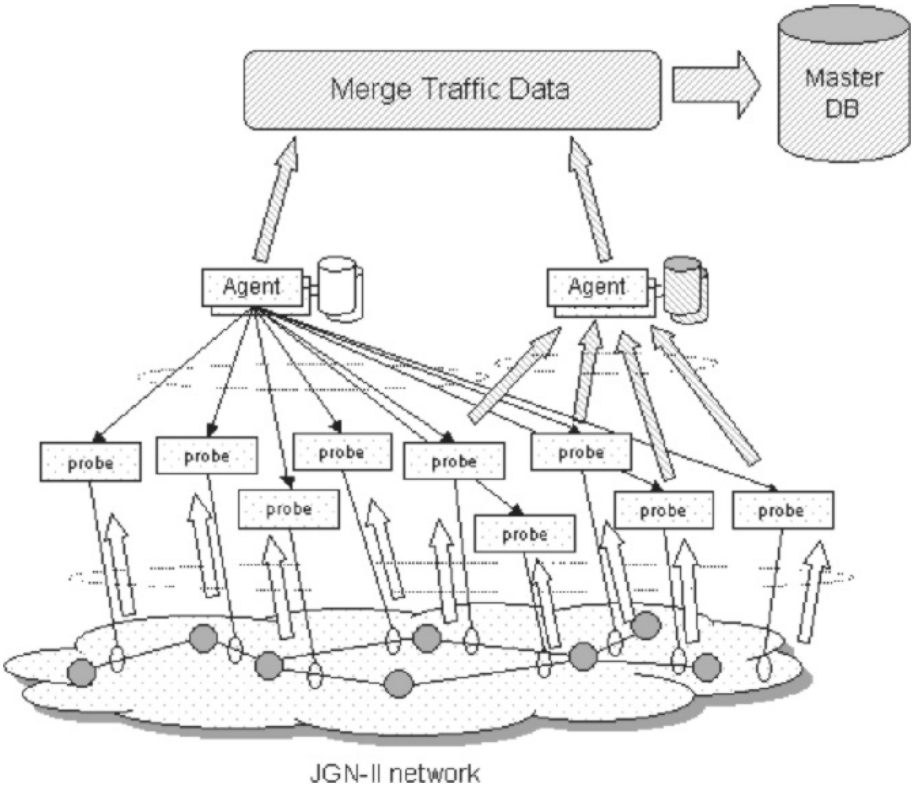

Fig. 3.1. Data Merge

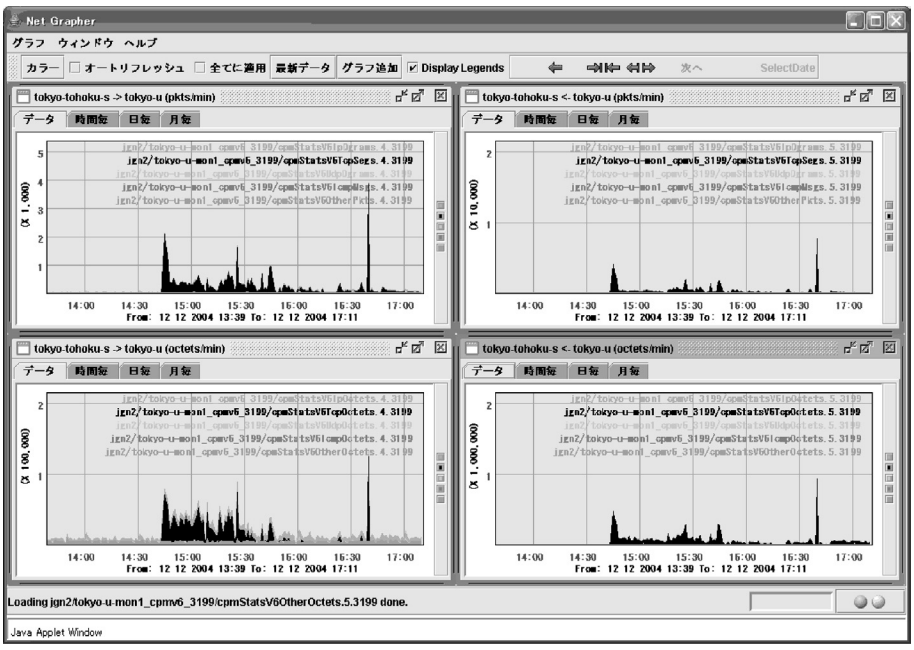

Fig. 3.2. Traffic Graph between RIEC in Tohoku University and the University of Tokyo

traffic volume between Research Institute of Electric Communication in Tohoku University and the University of Tokyo on Figure 3.2.

In our project we have provided an easy to use graphical user interface where the user can view the desired traffic by a few mouse-clicks (Figure 3.3).

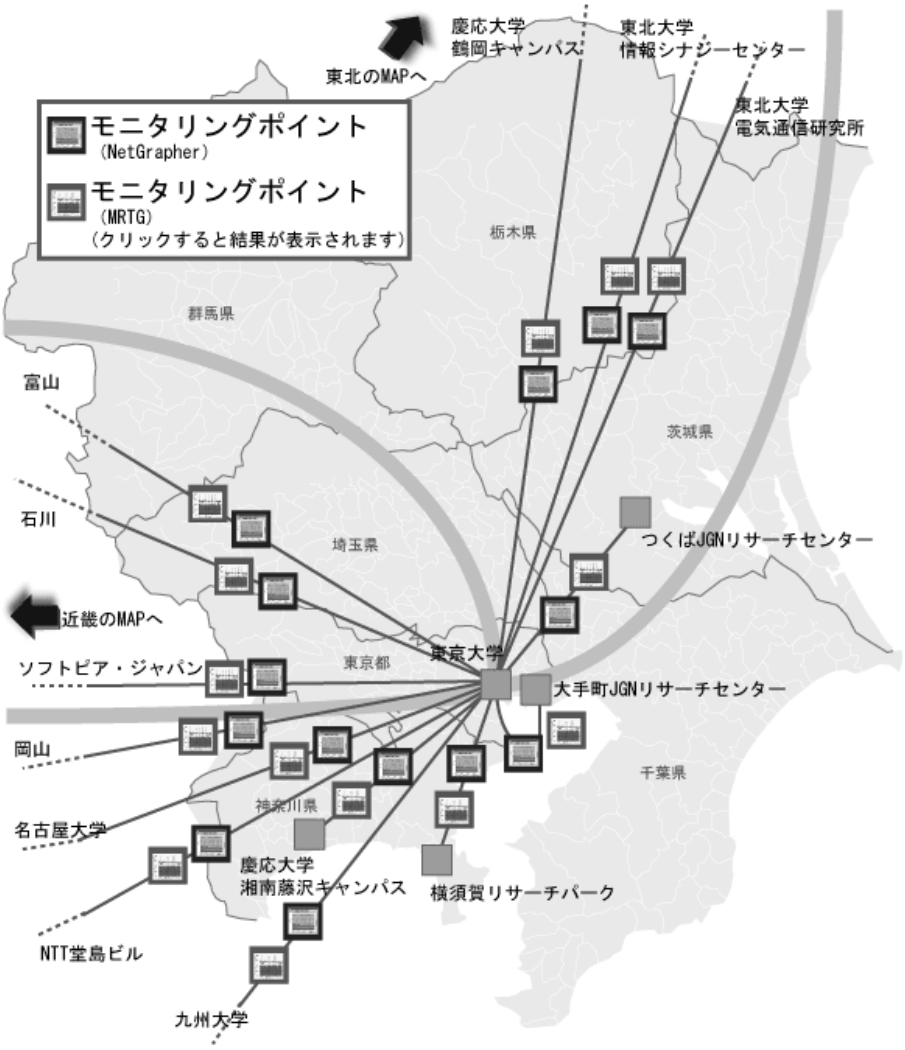

Fig. 3.3. Clickable Map to show network traffic

3.6 Latency

Network latency is one of the important indicators of network operational status. It may be used to evaluate quality of service and to estimate throughput for network application. We focus Round Trip Time (RTT) to examine latency.

There are many tools to measure RTT, such as ping, traceroute, skitter, pchar etc. RTT represents different statistics for each of these tools. In the following, we clarify the definition of RTT.

Figure 3.4 shows the path of a packet from one application to another across the network. The RTT from node A to node B along single path may represented by $\mathrm{RTT}_{\mathrm{A}\to\mathrm{B}}$ as

$$\mathrm{RTT}_{\mathrm{A}\to\mathrm{B}} = \delta_{\mathrm{A}\to\mathrm{B}} + \delta_{\mathrm{B}\to\mathrm{A}} + \delta_{\mathrm{Proc}}(\mathrm{B}).$$

The first term ($\delta_{\mathrm{A}\to\mathrm{B}}$) stands for the time which packet takes from Node A to Node B, and the second term ($\delta_{\mathrm{B}\to\mathrm{A}}$) vice versa. The last term ($\delta_{\mathrm{Proc}}(\mathrm{B})$) denotes the time taken at Node B to process the received packet and return a response.

If the application to measure the RTT is changed, the change will be reflected in latency. It is easy to imagine if we measure the RTT for a ping command and for and HTTP get command, the results will be very different. In our work, we examine two different RTTs. The first is the round trip time measured by traceroute6. Traceroute6 is the IPv6 version of

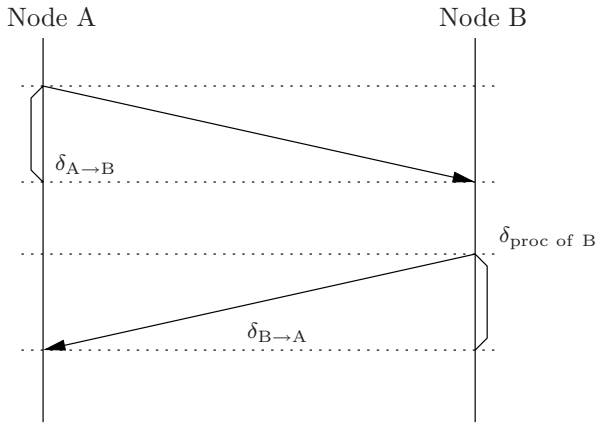

Fig. 3.4. diagram to measure Round Trip Time

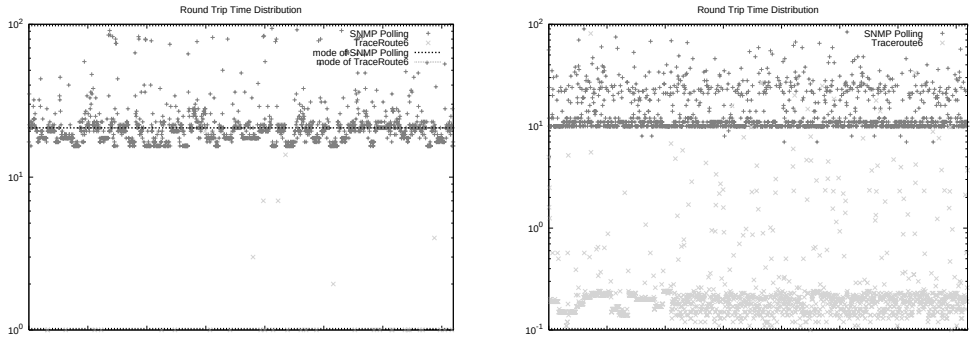

Fig. 3.5. Distribution of traceroute6 and polling interval for the node on the same LAN

traceroute. It sends UDP packets over IPv6 protocol. The payload length is 20 bytes. The IPv6 header is 40 bytes. It controls the "hop limit" field in the IPv6 header and attempts to elicit an ICMP6 TIME_EXCEEDED IN-TRANSIT and finally obtains an ICMP6 PORT UNREACHABLE response. This method measures the time interval from the instant the packet is sent to the instant an ICMP6 PORT UNREACHABLE packet is received.

The second RTT measure is the time interval from the instant the snmp get request is sent to the instant a response is received. We will call this the SNMP-RTT. We have been collecting these values every 60 seconds.

Firstly we examine the RTTs between two nodes on the same link. Figure 3.5 shows these values measured on 19 Nov. 2004. The left side figure shows the distribution of RTT (traceroute6) and the SNMP-RTT in Sendai. The right side figure shows the corresponding distribution in Kyoto. The X-axis shows the sequence number of the trials. The Y-axis scale is logarithmic. The nodes in this experiment connect on the same LAN. These are basically FreeBSD machines.

Figure 3.6 shows the distribution of RTT measured by traceroute6 and SNMP Polling interval between two different locations on 19 Nov. 2004. Left side figure shows from Sendai to a probe located at RIEC in Tohoku University. Right side figure shows from agents at Kyoto University to the same destination at RIEC.

Finally, we consider the mode and mean value of RTT by traceroute6 and SNMP polling interval. We plot the mode and mean value of traceroute6 and SNMP polling interval on a daily basis for the month of November 2004. Left side of Figure 3.7 shows the distribution of RTT (traceroute6) and Polling interval from Sendai to a probe located at Research Institute of Electrical Communication (RIEC), in Tohoku University is shown in the left hand figure. The right hand figure

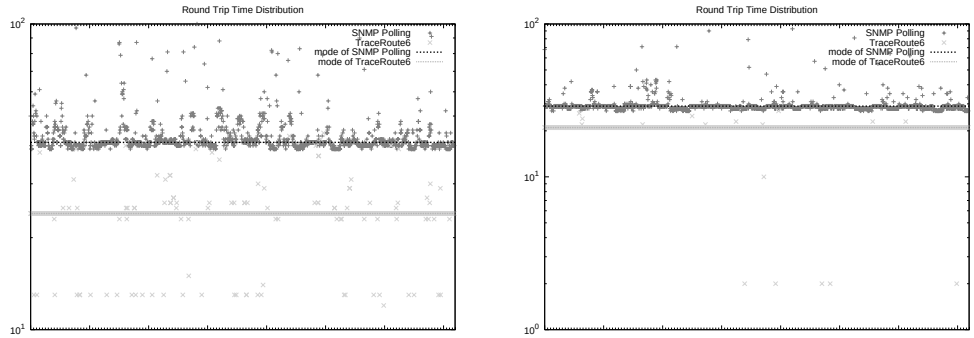

Fig. 3.6. Distribution of tracertoute6 and SNMP Polling to probe at RIEC

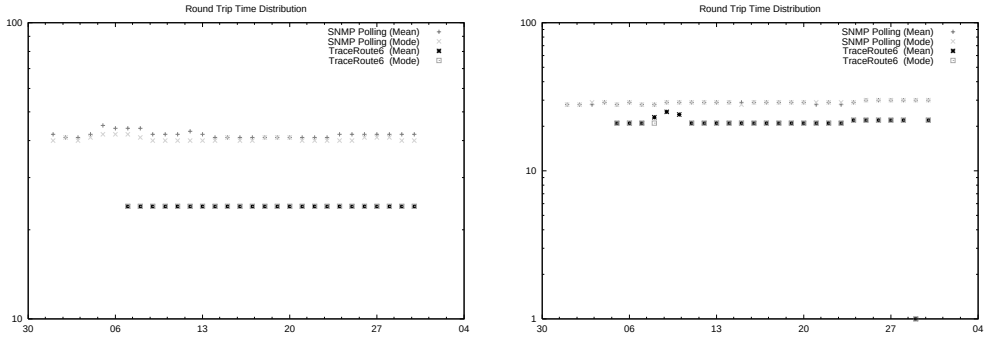

Fig. 3.7. Distribution of RTT and Polling interval to Tohoku Univ. from two different agents

Figure 3.7 shows the corresponding values from agents at Kyoto University to the same destination at RIEC. We can see from Figure 3.7 that there is no major fluctuation in the RTT of traceroute6 and SNMP polling interval.

3.7 Event-oriented network analysis

Here we discuss event-oriented network analysis. Event-oriented network analysis involves modeling network traffic statistics. Deviation from the established traffic model may be considered to be an indication of a network event.

For example of in the case of traffic volume, we explain a simple procedure which may be used by an event processor to detect probable indications of events[1]. We assume network traffic follows some statistical distribution e.g. Gaussian, Poisson etc. We evaluate the delta between traffic value estimated from previous data and the actual traffic value.

We show one of example using the simple method of moving averages. We attempt to estimate the next traffic value from the previous average and the deviation of previous N measured values. We put x_t as the measured traffic value at time t. Then we calculate the average μ_t and deviation σ_t of previous N at time t as

$$\mu_t = \frac{1}{N}\sum_{j=1}^{N} x_{t-j}, \quad \sigma_t^2 = \frac{1}{N}\sum_{j=1}^{N}(\mu_t - x_{t-j})^2. \tag{21}$$

For normalization purposes, we transform the measured value x_t to z_t as follows:

$$z_t = \frac{x_t - \mu_t}{\sigma_t} \tag{22}$$

This normalization transform lets us define the threshold for z_t. We say that an event has occurred at time t if z_t is beyond the threshold value.

Figure 3.8 shows traffic graph when network equipments are replaced. Our event detection algorithm using moving average regards it as event.

We also show Figure 3.9 as an example on the latency analysis. This figure shows the same charts of Figure 3.6. But the destination in this case is a probe located at the University of Tokyo.

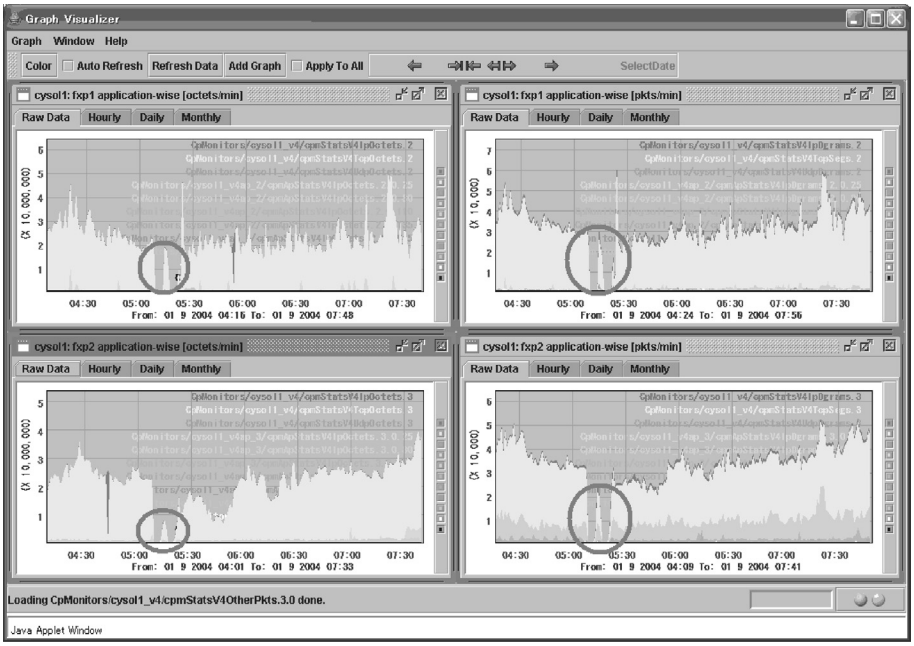

Fig. 3.8. Traffic Graph in replacing network equipments

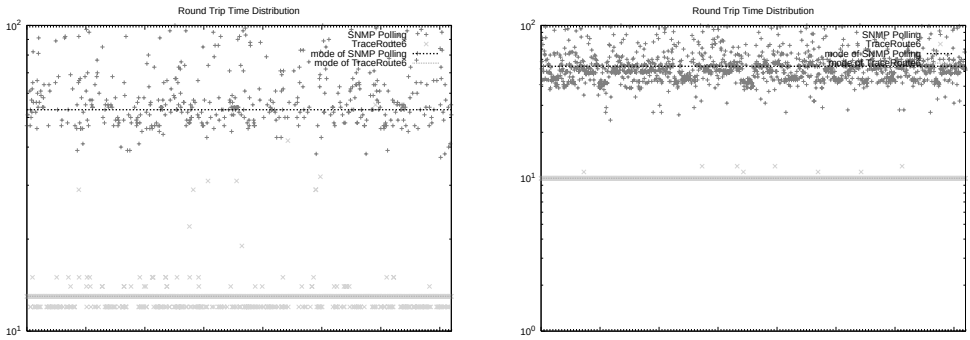

Fig. 3.9. Distribution of RTT and Polling interval to Univ. of Tokyo from two different monitors

This figure shows the median value and mode value of traceroute6 are stable, but there is a variation in the values of SNMP-RTT when compared with the corresponding values at the RIEC probe. The case of Sendai shows the variation very clearly.

We can say that one of the reason for the fluctuation in SNMP-RTT for the University of Tokyo is that the probe has been monitoring as many as 19 links. The probe probably lacks the ability to deal with monitoring so many links.

3.8 Conclusion

In this chapter, we introduce our monitoring and analysis activities. About monitoring activities, we show our environment in the JGNII network.

About analysis activities, we show our monitoring items, one is traffic volume and the other is latency. We also discuss event detection with these statistics applying for network management.

We plan to study the following as future work: We will estimate the accuracy of detections of indications of events. We will also evaluate the suitability of other traffic models to detect events. We will investigate the area of event classification, for example the relationship between indices.

第4章 Issues about merging network data collected in a redundant environment

4.1 Introduction

Network Traffic Monitoring is an important aspect of network management and security. For example, we might observe the effects on the network traffic when an event, such as a network failure, an operational failure or a security incident has occurred. And we would know the predictive information about quality of service, throughputs and also. We, netman WG, have been collecting traffic statistics for the JGNII network to study the results.

In this monitoring, we put two collecting agents to secure verbosity. But we adopt SNMP framework with IPv6 network to collect traffic statistics, so we cannot assure of the identity of these collected statistics.

The purpose of this document is to point out issues of multiple collecting agents with SNMP and we will suggest the solutions.

4.2 Environment

Before discussing problems, we start to introduce our monitoring environment in JGNII.

JGNII is an open test-bed network environment for research and development and provides nationwide IPv6 network and optical wavelength networks in Japan.

We project a network traffic monitoring in JGNII network. We aim at providing user network traffic information and analysis techniques to use for research and experiment in JGNII network. We have been placing probes at Miyagi, Tokyo, Gifu, Kyoto, Hiroshima and Saga, and polling traffic statistics from agents on Sendai and Kyoto.

Table 4.1 shows our monitoring environment as on 26 July 2004. We place probes monitoring lines passively at all points in JGNII. We use SNMP framework to collect traffic statistics. Two Polling Agents at Sendai and Kyoto periodically, every 60 seconds, access probes by SNMP over IPv6 and obtains traffic statistics which is obtained in the form of Managed Objects. We open these traffic information to the public at `http://www.cysol.co.jp/research/jgn2mon/`.

Table 4.1. Monitoring Environment in JGNII

Items	Number
Sites where probe is placed	9
Placed probes	10
Monitoring points	11
Monitoring links	26
(with VLAN)	(19)
Polling Agents	2

Table 4.2. Measuring statistics

IPv6 packets/traffic volume
ICMPv6 packets/traffic volume
TCP over IPv6 packets/traffic volume
UDP over IPv6 packets/traffic volume
Other protocols packets/traffic volume
SNMP Polling Interval
Elapsed time by Traceroute6

We show our monitoring traffic statistics on table 4.2. Here “other protocols” means that it is an IPv6 packet but its next header field is not ICMPv6, TCP or UDP. We also have been collecting Round Trip Time measured by traceroute6.

4.3 Issues

From previous section, we show that we place two polling agents and they collect network traffic periodically, every 60 seconds. The reason that we adopt two different polling agents is for data redundancy. But this matter causes other issues. We show an example table to point out what kind of matter this data redundancy causes here.

From table 4.3, we can clarify three issues:

A. Timestamp issue

Timestamp of every polling is generated at each polling agent, so, those are fluctuated due to:

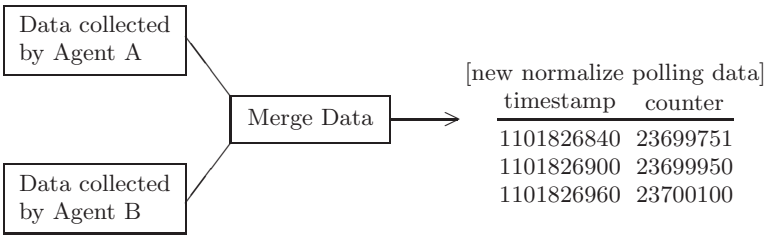

Fig. 4.1. Image about plans for solutions

Table 4.3. example issue caused by redundancy of polling agents

Polling Agent A		Polling Agent B	
timestamp	counter	timestamp	counter
1101826839	23699750	1101826811	23697998
1101826901	23699954	1101826880	23699902
1101826962	23700101	1101826940	23700008
1101827021	23700119	1101827000	23700119
1101827081	23700263	1101827060	23700219
1101827141	23700840	1101827120	23700458
1101827200	23700942	1101827180	23700894
1101827260	23701050	1101827240	23700996
1101827320	23701163		
1101827380	23701271	1101827360	23701209

- network distance from each polling agent is different
- no synchronization mechanism between two polling agent

B. Polling interval issue

Polling interval is also fluctuated by configuration and load of each polling agent. So, it is not easy to compare them.

C. Data lack issue

We use SNMP, async protocol, to collect traffic statistics from probes with UDP over IPv6 protocol which does not have the responsibility of delivering packets.

Those are reasons that we cannot guarantee the sameness of the contents in two different polling agents. So it is necessary to solve these issues.

4.4 Plan for solutions

We discuss here how to solve issues which we point out at Sec. 4.3. Fig. 4.1 shows out plans to solve these issues.

First we decide the master traffic statistics by comparing the data polling agent has. We consider the parameters to decide the master are number of successful polling times, mean of polling interval and variance of polling interval. Next step is that we complement the traffic statistics which master does not have. Finally, if there is a fluctuation in polling intervals, then we normalize the interval with the traffic statistics allocating by pro rate.

Again, we enumerate our plan to merge traffic statistics below:

a. Decision of the master polling agent for merging traffic statistics.
b. Complementation from other polling agent
c. Timestamp normalization by pro rate allocation

To solve these issues, we are studying on merging traffic statistics which two different polling agents collect.

4.5 Conclusion

Actually our network monitoring project faces with an issue such that there is no guarantee the sameness of the contents in two different polling agents. In this document, we point out an issue caused by the multiple polling agents and make a plan to solve. As future work, we are studying on and we are implementing merging traffic statistics which two different polling agents collect.

第 5 章　CpMonitor リモートアップデートツールの開発

5.1 動機

netman WG では、JGNII IPv6 ネットワーク上において、複数のパッシブ型トラフィックモニタ

（CpMonitor）を各地に配備し運用を行ってきた。CpMonitor は実運用から得られたノウハウをフィードバックしながら機能の拡張を頻繁に行ってきている。

CpMonitor の機能拡張にともなう各地に配備されたモニタ装置のバージョンアップ作業は、従来運用チームが手分けをして人手で行っていた。しかし、その更新頻度の高さとモニタ装置の台数の多さにより、以下のような問題が無視できなくなってきた。

(1) バージョンアップ作業による観測不能時間の拡大
(2) 人為的作業ミスによる観測不能項目の発生
(3) 管理者作業負荷の増大

そこで、これらの状況を改善するために、従来人手で行ってきた作業をすべて自動化すべく、CpMonitor Remote Update System（RUS）の開発を行った。

5.2 方針

RUS の開発にあたって、先に挙げた問題を改善するために、以下の要求項目を満たす機能を実装することとした。

(1) 更新作業時間の短縮
(2) 更新作業後の更新確認作業の自動化
(3) 安全な更新環境の実現

(1) については、以下に挙げる機能を実装することとした。

(a) 複数台のモニタ装置に対する、CpMonitor アプリケーションの一括更新機能
(b) 更新後の自動再起動機能

また、(2) については

(a) SNMP アクセスによる自動バージョン取得機能

を実装することにした。さらに (3) については

(a) マネージャ装置およびモニタ装置間の認証機能
(b) マネージャ装置およびモニタ装置間の通信の暗号化

を実現することとした。

5.3 実装

RUS は、各モニタ装置に CpMonitor のソースコードが展開され、コンパイルおよびインストールが完了し、CpMonitor 自体の設定と実行が行われている環境を前提として実装している。

RUS の実装にあたっては、一般的な UNIX マシンで利用可能な以下のツール類を用いた。

(1) 使用言語：Bourne Shell
(2) 利用フリーソフト：OpenSSH・sudo・net-snmp

実装した RUS は 2 つの Shell script により構成される。

(a) updateForceCpMonitor.sh
(b) updateForceCpMonitorBackend.sh

これらのスクリプトはマネージャ装置上に置いてあり、バージョンアップの対象となるモニタ装置のリストはあらかじめ (a) のスクリプト上に記述しておく。

更新作業を行うためには、CpMonitor アプリケーションにおいて変更があったファイルを、ソースツリーにのっとってあらかじめ tar で固めておく。その後、その tar ファイルのファイル名を引数とし、(a) のスクリプトを起動する。

（例）`./updateForceCpMonitor.sh patch-0902.tgz`

(a) のスクリプトが起動されると、以下の操作が行われる。

(i) (b) のスクリプトの転送
(ii) あらかじめ用意された tar ファイルの転送と展開
(iii) モニタ装置上での (b) のスクリプトの実行
（い）ソースコードのコンパイルとインストール
（ろ）起動されている CpMonitor の停止
（は）更新された CpMonitor の起動
(iv) モニタ装置上の (b) のスクリプトの消去
(v) SNMP アクセスによる CpMonitor のバージョン確認

5.4 環境設定

各モニタ装置上に、RUS で利用するための、以下の条件を満たしたユーザアカウント “cpmupdate” を作成する。

(1) ログイン認証には、スクリプト実行者の RSA ないし DSA 鍵を用いる
(2) パスワードなしで sudo により特権を利用できる権限を持つ

具体的な操作は以下のとおりである。

(a)

```
$ sudo vipw
```

以下の行を追加。

```
---
cpmupdate:*:3999:3000::0:0:CpMonitor
   remote updater:/home/cpmupdate:/bin/sh
---

$ sudo mkdir /home/cpmupdate
$ sudo mkdir /home/cpmupdate/.ssh
```

表 5.1. updateForceCpMonitor.sh

```
#!/bin/sh

AGENTS="\
v6mon.nemoto.ecei.tohoku.ac.jp \
v6mon-trix \
corsa.shiratori.riec.tohoku.ac.jp \
pc4.sendai.wide.ad.jp \
tokyo-u-mon1 \
softp-mon \
kyoto-mon \
hiro-mon \
hiro-cu-mon \
saga-u-mon \
"

LOG=updateForceCpMonitor.log

if [ ! -f "$1" ]
then
  echo "$0: $1 is not found"
  echo "usage: $0 <CpMonitor patch filename>"
  exit 1
fi

touch $LOG

echo                                  >> $LOG 2>&1
date                                  >> $LOG 2>&1
echo "agents to be updated: $AGENTS" >> $LOG 2>&1

for agent in $AGENTS
do
  echo                                                                    >> $LOG 2>&1
  echo "[[ $agent ]]"                                                     >> $LOG 2>&1
  snmpget -Ov -v3 -lauthPriv $agent sysDescr.0 cpmMonitorVersion.1 >> $LOG 2>&1

  scp updateForceCpMonitorBackend.sh cpmupdate@$agent:                    >> $LOG 2>&1
  ssh -l cpmupdate $agent sudo tar -xvzf - -C /usr/local/src/CpMonitor5-SNAP < $1 >> $LOG 2>&1
  ssh -l cpmupdate $agent sh -x updateForceCpMonitorBackend.sh            >> $LOG 2>&1
  ssh -l cpmupdate $agent rm updateForceCpMonitorBackend.sh               >> $LOG 2>&1

  snmpget -Ov -v3 -lauthPriv $agent sysDescr.0 cpmMonitorVersion.1 >> $LOG 2>&1
done

echo >> $LOG 2>&1
date >> $LOG 2>&1
exit 0
```

```
$ sudo chmod 700 /home/cpmupdate/.ssh
$ sudo vi
   /home/cpmupdate/.ssh/authorized_keys

$ sudo chmod 600
   /home/cpmupdate/.ssh/authorized_keys
```

スクリプト実行者の SSH 公開鍵を登録する。

```
---
ssh-dss
AAAAB3NzaC1kc3MAAACBAI9PvMLdrbHNmnXN+tFtPi
p18dz+ntHbfq9QZkGpjwuKoVRjDrVBdObTcLGH1If7
6NGVPSb5IgxdzaMYCIjz7nw8Sww0TTH80iHBElmZUG
sAOJRirJ/QKT/Fx8zj18TuNb1jSYOeIpXo/GrLuHv2
z8+3CZHIHP7STNdpEtqAQZY5AAAAFQCh1si/p1SGci
5riBKU9Gk3UILOhQAAAIAo8o3TEjuKT2nmq7XFuTll
/W7EZQVdSnwr4WzVYqbqkUBFOBhR77MnNQx5KLQr0z
ig58dN/7snoOXCekJFq+SR2vROIBMEfb1SqxnFzLfx
GsCC8di1KFRQr2kqjQQlwDHLsaLQvn8RzJgJhigiVy
5EOcLv6+eApD/DtpHYcflBvAAAAIEAi19Dm+POkf9+
jlXT6LJvq1QhCZGJQlSicv5ahqYJK4gCdyZEK8Zg65
Y2UbiXh8R6qni8Zb82xnc4iYEuxs5stfuTU+ZfV2Nj
a7s4BOJ/Bp7W8haSTR8OomIxVlOZXGSpE2ep+0yYFi
j1FpxnrBF5jJ53vHHfGU4LORWyULx4CIc=
dsa-key-20020416
ssh-rsa
```

表 5.2. updateForceCpMonitorBackend.sh

```
#!/bin/sh
DATE=‘date +%y%m%d%H%M%S‘# yymmddHHMMSS

SRCDIR=/usr/local/src/CpMonitor5-SNAP/src
BINDIR=/usr/local/CpMonitor/bin
SBINDIR=/usr/local/CpMonitor/sbin

if [ -d "$SRCDIR" ]
then
  if [ -d "$SRCDIR/SnortForTaoIPv6" ]
  then
    cd "$SRCDIR/SnortForTaoIPv6" && sudo make all
    if [ $? -eq 0 ]
    then
      sudo mv "$BINDIR/snort" "$BINDIR/snort.$DATE"
      sudo make install
    fi
  fi

  if [ -d "$SRCDIR/net-snmp-5.0.9" ]
  then
    cd "$SRCDIR/net-snmp-5.0.9" && sudo make all
    if [ $? -eq 0 ]
    then
      sudo mv "$SBINDIR/snmpd" "$SBINDIR/snmpd.$DATE"
      sudo make install
    fi
  fi

  sudo sh -x /usr/local/etc/rc.d/CpMonitor5.sh stop
  sudo sh -x /usr/local/etc/rc.d/CpMonitor5.sh start
fi
exit 0
```

```
AAAAB3NzaC1yc2EAAAABJQAAAQBWPYJJEq0qsi0Q/u
g26sgstdsbWmv3nDqwvPt1YUDKdpJgWLD4vro93ni8
doZbLDd1B3SbFcWxUhZVkKyDvTW+IjVbSuHeY4Wvwu
lcm+p4OFVxA5s3b/3rIefrrojbMNjKboqQnndqgHeP
qVsF/FBv4xI3/bh7XHKO0EHzKafSFDTgRoEPkc7Q0p
TQWfJlc/8OuMZCMB3p2ga17h7e1xqTZU+Yw7mtm0AL
xadiE2gaWeoWU1x8IBzDKAZEHldQOpd5rNVjQ83HPl
KlkmvrQqdUKckoO6Uq5Paf3qhpPcDAYSJK9HE4HRQB
NxZwVHGGknDsjgsgw52IIAZ9EoqZG6kR
rsa-key-20021210
---
$ sudo chown -R cpmupdate /home/cpmupdate
```

(b)

```
$ sudo visudo
```

以下の行を追加。

```
---
# cpmupdate can use root priviledge
   without password (040930)
cpmupdate   ALL=(ALL)   NOPASSWD: ALL
---
```

5.5 現状と今後

先に述べたとおり、RUS を実行するためには、あらかじめ CpMonitor の更新ファイルを tar ファイルとして準備し、以下の様にコマンドラインから実行する。

```
./updateForceCpMonitor.sh patch-0902.tgz
```

これにより、各モニタ装置上の CpMonitor のソースツリーにパッチが適用され、新しいバイナリの生成とインストール、古いバイナリのバックアップが行われる。その後に CpMonitor の再起動とバージョン情報の収集が行われる。これら一連の作業ログは $LOG（標準では updateForceCpMonitor.log）に保存される。

JGNII モニタリング環境において、RUS を用いた 9 台のモニタ装置のアップデート作業を行った結果、25 分ですべての装置上のアプリケーションのバージョンアップ作業とバージョン確認を行うことができた。

今後は以下に挙げる機能の実装と拡張を予定している。

(1) CpMonitor 新規インストールへの対応
(2) エラーチェックの厳密化