5 Application

5-1 A Report on the Experiment of Combined Operation via JGNⅡ

MIWA Shinsuke and OHNO Hiroyuki

To analyze security incidents and verify security measures, an environment is required that enables the user to track and analyze phenomena caused by attacks, while taking the interaction among systems into consideration. However, security incidents have recently been growing more and more complex due to an increase in scale, simultaneous incidents, or interactions among multiple events.

As a result, it is becoming increasingly difficult to reproduce security incidents within a single experimental environment. We will discuss the combined operation of existing experimental environments at minimum cost as a way to handle security incidents of increasing scale and complexity.

In this paper, we report on the connecting experiments of our reproducing environments called “SIOS”, “VM Nebula” and “StarBED” via JGNⅡ.

Keywords
Internet, Security, Security incident, Experimental environment, Incident reproducing

1 Introduction

An enormous number of security incidents take place on the Internet every day. While a variety of security measures are planned and implemented against such incidents, these measures are always countered by new methods of attacks, thus requiring continuous development of new security measures to resolve emerging security threats. Accordingly, we are facing an urgent need for a new security technology that will offer a radical solution to this problem.

When developing new security measures, policies, or technologies, we must be able to evaluate effectiveness and to determine whether they result in any adverse side effects. To this end, we conducted R&D on an experimental environment to reproduce security incidents, one that can serve as a platform for the evaluation of unauthorized access, attacks, and other security incidents.

Security-incident reproduction capabilities (involving features such as the extent and accuracy of reproduction) and the degree of traceability vary greatly depending on the type of experimental environment used. Generally, if an experimental environment is highly accurate in reproducing security conditions, it tends to offer poor traceability, and the environment is difficult to operate. If on the other hand an experimental environment results in theoretical reproduction, the degree of traceability will be high, enabling easier implementation of the experimental environment.

Many security incidents are triggered by specific vulnerabilities in the running of the OS or of the applications. Therefore, an analy-

sis of causes requires an environment that can reproduce specific vulnerabilities. Further, to verify the effectiveness of a countermeasure requires conformance tests based on implementation. Moreover, the range of effects of security incidents has also been expanding in recent years, and many new security technologies are designed for wide-range application. Therefore, a large-scale experimental environment is essential for analysis and evaluation.

As discussed above, the experimental environment used to investigate security incidents must enable testing based on actual implementation. In other words, the experimental environment must offer high reproducibility with respect to such incidents. However, due to the expanding range of incident effects and a widening range of countermeasure technologies, a high degree of traceability is also demanded. To respond to these requirements, a single experimental environment can no longer provide sufficient performance in the reproduction of security incidents.

Given this problem, we connected different existing experimental environments of different types and used them in combination with the aim of improving reproduction and degree of traceability while keeping experimental costs to a minimum. This paper reports on the combined operation of existing experimental environments to reproduce the types of security incidents we are seeing today, reflecting an increasing scale and growing complexity.

Specifically, this report describes an integrated environment created for the study of the combined operation of multiple experimental environments. We connected SIOS (Security Intelligent Operation Studio), VM Nebula, and StarBED, three experimental research environments we have used in the past, via JGNⅡ. Below we report on the results of our experiments with a special focus on an experiment for connection of VM Nebula and StarBED.

2 Outline of the individual experimental environments

The following presents an outline and characteristics of each of our experimental environments designed to reproduce security incidents.

2.1 SIOS (Security Intelligent Operation Studio)

The SIOS (Security Intelligent Operation Studio) designed to reproduce and test security incidents[1] involves an experimental environment created with actual equipment; this environment was designed to test Internet security issues, and is part of the overall system referred to as SIOS[1]. This equipment is installed at the Koganei Headquarters of the National Institute of Information and Communications Technology in Koganei City, Tokyo.

This SIOS consists of a DDoS-attack simulation section consisting of 100 PCs, an Internet section featuring bandwidth control, and a "victim" section equipped with various firewalls, IDS, and DNS/SMTP/web servers (Fig. 1).

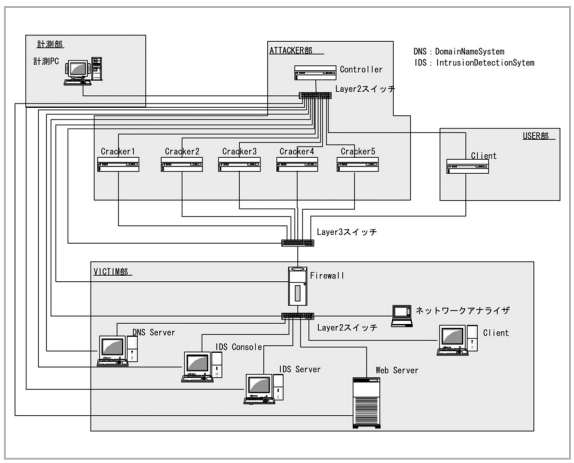

Fig.1 *SIOS (Security Intelligent Operation Studio)*

The structure of the test system has recently been made more flexible. The 100 nodes in the attack simulating section can not only reproduce attacks but also serve as a "victim" section or Internet section, since specific functions can be assigned to the individual nodes.

Using actual attack tools, simulated DDoS attacks can be launched by as many as 100 nodes at the same time, thus allowing for simulation of the effects of actual attacks.

Further, agents for the management of experiments and measurements are also used for automated execution of experiments. Although the system involves a large-scale experimental environment using actual equipment, this automation enables detailed management of experiments and reduces operator burden.

1 SIOS stands for Security Intelligent Operation Studio. The basic system was developed jointly by the Emergency Communications Group of the Communications Research Laboratory (presently the National Institute of Information and Communications Technology) and Yokogawa Electric Corporation. SIOS is a product commercialized originally by Yokogawa based on this system, and SIOS is a registered trademark of Yokogawa Electric Corporation.

2.2 VM Nebula

VM Nebula [2] is an experimental environment created by a PC emulator. This system is designed to combine high accuracy in reproduction (through an experimental environment employing actual equipment) with the flexibility and scalability provided by a simulated experimental environment. This environment is installed in the Kansai Advanced Research Center of the National Institute of Information and Communications Technology in Kobe City, Hyogo.

The VM Nebula system is comprised of four mock servers serving as PC emulators and two multilayer switcher units used for physical connections among the servers (Fig. 2).

All servers are identical, and each of the two multilayer switcher units is connected to all servers. Since all servers are capable of identical functions, their roles are interchangeable—each of these roles and functions can be assigned to any one of the four servers.

Attacker PCs and victim PCs are simulated using virtual PCs created by the PC emulator. Firewalls, IDS, and victim sites on servers are simulated using multiple virtual PCs. The

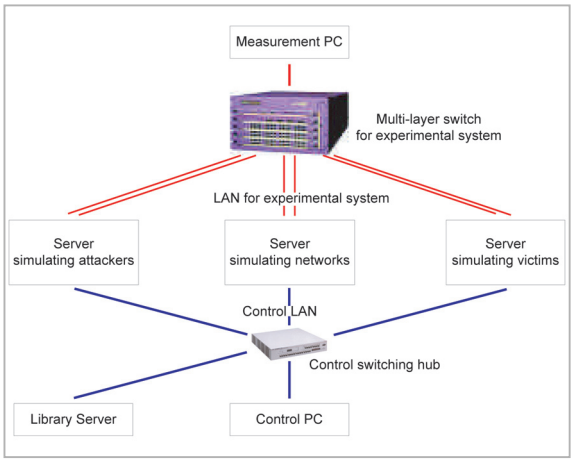

Fig.2 VM Nebula

Internet is simulated by connecting VLANs through multilayer switchers and limiting bandwidth, with the additional use of a virtual PC to execute routine software as a virtual router.

The system enables actual OS and server implementation as well as the use of PC-operable attack tools without modifications.

Since the VM Nebula can store and distribute the configuration of virtual PCs and settings for multilayer switches, once an experimental system is set up, it can be easily reconstructed at any time, thus enabling the analysis [3] of viruses and worms that require frequent retesting. It should be noted, however, that this system is not equipped to automate experimental-execution procedures.

2.3 StarBED

StarBED [4] is the name given to a facility for large-scale network experiments installed in the Hokuriku IT Open Laboratory of the National Institute of Information and Communications Technology in Nomi City, Ishikawa Prefecture.

StarBED consists of 512 PCs and the switches that connect them. Each PC has two or more network interfaces, which are connected to the experimental network and management network. This setup is essential for separating experimental traffic from management traffic, and it prevents unexpected traffic from affecting the results of experiments. The

management network is provided with a file server and a DHCP server to facilitate experimentation, and also features nodes that operate the applications we have developed for the automatic establishment of an experimental environment and automatic execution of scenarios (Fig. 3).

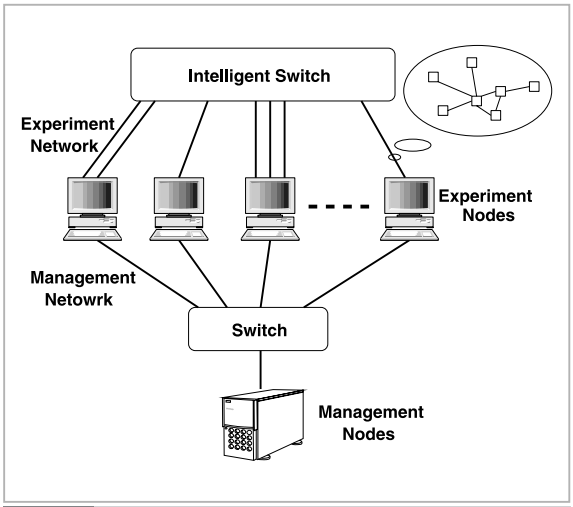

Fig.3 StarBED

An experimental topology can be constructed by changing the switch settings, without having to alter the physical topology of the nodes or switches. This enables simultaneous use by multiple users and reduces the cost of constructing an experimental topology. The user can create the necessary experimental topology by changing the settings of the switches that hold the simulated network interfaces for the nodes.

Using tools to support experimentation, the user can automate the construction of an experimental topology and the execution of the experiment by simply entering experimental settings.

3 Integration experiment

As a preliminary integration test, we carried out an experiment to assess the performance of the connection environment. The following describes the details of this experiment.

3.1 Connection environment

JGNII was used for physical connection of three experimental environments. This setup was selected for the following three reasons.

- The setup provides broadband (10-Gbps) network connectivity.
- The separate network formed is isolated from the Internet using dedicated experimental circuits.
- All three experimental environments are close to JGNII access points, thus enabling the establishment of connection environments at low cost.

For actual connection, JGNII's "multipoint simultaneous connection service" was used to connect three points via Ethernet, with connections switched[2] among multiple VLANs according to the needs of the experiment. The connection point in Koganei City in Tokyo was linked to the connection point in Nomi City in Ishikawa Prefecture by a 10-Gbps line, while the connection to the point in Kobe City in Hyogo Prefecture took place over a 1-Gbps line.

Our experiments assumed the following applications and use with respect to the individual VLANs. We prepared four VLANs in a standard configuration—three VLANs performing the following functions and one spare VLAN.

- operation, control, and performed measurement of experimental environments
- mutual remote operation
- communication between experimental nodes

It was necessary to adjust IP addresses and others used for operation management with a layer-2 connection, which does not perform routing.

2 Since this type of service was not previously offered, it was specially set up for our experiments. VLAN switching was performed manually.

3.2 Connection experiments

To conduct a connection experiment, we connected VM Nebula and StarBED using the VLAN for operation, control, and performed measurement of experimental environments. Figure 4 shows the connection diagram.

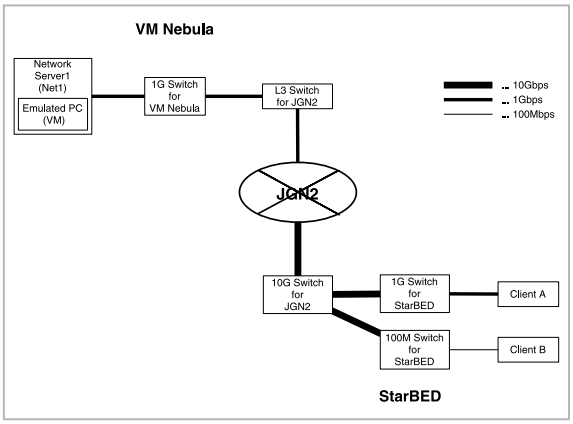

Fig.4 *VM Nebula and StarBED connection diagram*

Table 1 shows the NICs and OSs of the nodes used in the experiments.

All nodes used in the experiment were given private IP addresses within a single network.

To evaluate connection performance, we used the ping command to measure RTT and used netperf [5] to measure TCP_STREAM and UDP_STREAM performance. Table 2 summarizes the results. Note that TCP_STREAM and UDP_STREAM corresponded to throughput, with units of Mbps (10^6 bits/sec), while the RTT values are average figures obtained from transmission of ICMP 100 times using the ping command; these values are given in ms.

As the table indicates, the throughput of TCP_STREAM between VM Nebula and StarBED was approximately 1/12 of maximum, which was abnormally low. In the individual experimental environments, TCP_STREAM was the same or faster than UDP_STREAM. Given these observations, it is possible that an abnormality arose when connection was established via JGNⅡ. Our investigation indicated that this abnormality was due to an LFN problem[3].

In UDP_STREAM, the throughput between Net1 and A, connected by a Gigabit Ethernet interface, was approximately 400 to 550 Mbps. This speed is considered to correspond to the performance limit, based on the processing performance of the nodes and connection via JGNⅡ. For B, which was connected via 100Base-TX interface, throughput was limited to approximately 90 to 95 Mbps between any given nodes. We believe that this represents the performance limit, based on network interface capacity.

In contrast, the throughput between VM and A was approximately 100 to 150 Mbps. This is nearly equivalent to the internal performance of VM Nebula, and considered to be the limit imposed by the performance of the virtual PC software and that of the server operating this software.

3 This refers to the so-called "Long Fat Network" problem, a phenomenon in which transmission performance is restricted by window size and RTT when the TCP window is smaller than the bandwidth delay product (i.e., capacity) of the circuit. In recent TCP implementation, this problem has been circumvented using a window scaling option. The OS used in our experiments was also equipped with this option, but for some reason the option failed to function. We are currently investigating the reason for this failure.

3.4 Large-scale attack-simulation experiment

Next, we conducted an experiment to simulate large-scale attacks, using three experimental environments to reproduce security incidents—SIOS, VM Nebula, and StarBED—connected via JGNⅡ.

The experimental procedure was simple, as described below.

(1) F5 attacks were launched to disable service of the target web server.
(2) A mobile filter was used to restore this service.

F5 attacks have been frequently used in recent cyber experiments. These attacks involve user activation of the F5 function key of Internet Explorer when the program displays a designated webpage at a certain time (announced on an electronic bulletin board or otherwise). The "Reload this page" function is assigned to the F5 key. When many people press this function key at once, a large amount of HTTP requests are sent to the server providing the designated webpage. F5 attacks are designed to cause flooding—i.e., an overflow in the downlink to the server caused by a very

Table 1 *Experimental nodes and corresponding NIC and OS*

Environment	Node	Abbr	NIC	OS
VM Nebula	Network server 1	Net1	1000Base-SX	RedHat Enterprise Linux AS 3.0
	Virtual PC node	VM	1000Base-SX	FreeBSD 5.3R
StarBED	Client A	A	1000Base-T	FreeBSD 5.3R
	Client B	B	100Base-TX	FreeBSD 4.4R

Table 2 *Experimental results*

Sender	Receiver	TCP_STREAM	UDP_STREAM	RTT
VM Nebula intra				
Net1	VM	159.90	137.39	0.252
VM	Net1	298.90	152.30	0.252
VM Nebula→StarBED				
Net1	A	45.14	549.36	8.711
Net1	B	15.32	95.78	8.668
VM	A	28.40	152.63	10.580
VM	B	15.08	95.54	11.007
StarBED→VM Nebula				
A	Net1	24.25	412.70	8.754
A	VM	20.10	117.98	8.925
B	Net1	14.38	95.75	8.766
B	VM	14.28	93.24	8.747
StarBED intra				
A	B	93.24	95.88	0.228
B	A	93.38	95.93	0.211

high volume of requests, exceeding the allowable number of requests, and an overflow in the uplink from the server due to the responses triggered by requests. F5 attacks represent a rather primitive DDoS technique; they do not falsify the IP address.

"Mobile filter" techniques generally describe methods used to move a filter application point. In the case of bandwidth-consuming DDoS attacks, even if packets are discarded by the firewall located at the entrance/exit of an organization's network, a lack of bandwidth can arise at this entrance/exit of the organization; therefore, this packet discarding behavior serves no purpose. Therefore, an upstream external organization such as the provider is usually asked to filter out packets at a location closer to the transmitting source. The mobile filter technique moves the filter application point automatically to a location near the source of attack by combining an IP traceback technology with a filter; this function currently must be executed manually. Several systems have been proposed, but none has been realized to date. In our experiment, the filter moves according to the scenario description, and is not a mobile filter in a true sense.

The experimental scenario called for the issuance of attacks from SIOS to the server (the "victim") on VM Nebula via routers on StarBED; details are as follows (see Fig. 5).

- Approximately ten attacker nodes in SIOS were used to repeat transmission of the HTTP request (simulation of F5 attacks).
- To ensure that all attacks reached the destination through different routes, approximately 50 routers were arranged on StarBED (simulation of realistic Internet delay).
- The firewall and web server were constructed on VM Nebula as separate virtual PCs (simulation of a target website).

The SIOS attacker nodes applied the automatic experiment-execution function to repeat the HTTP request transaction to simulate attacks. We set up 15 attacker nodes to execute these transactions all at once.

On StarBED, we constructed 54 PC routers using Zebra routing software. To simulate attacks from multiple routes, the number of hops in the route was varied for each node. In addition, some PC routers were added with a script to launch the "ipfw" firewall software,

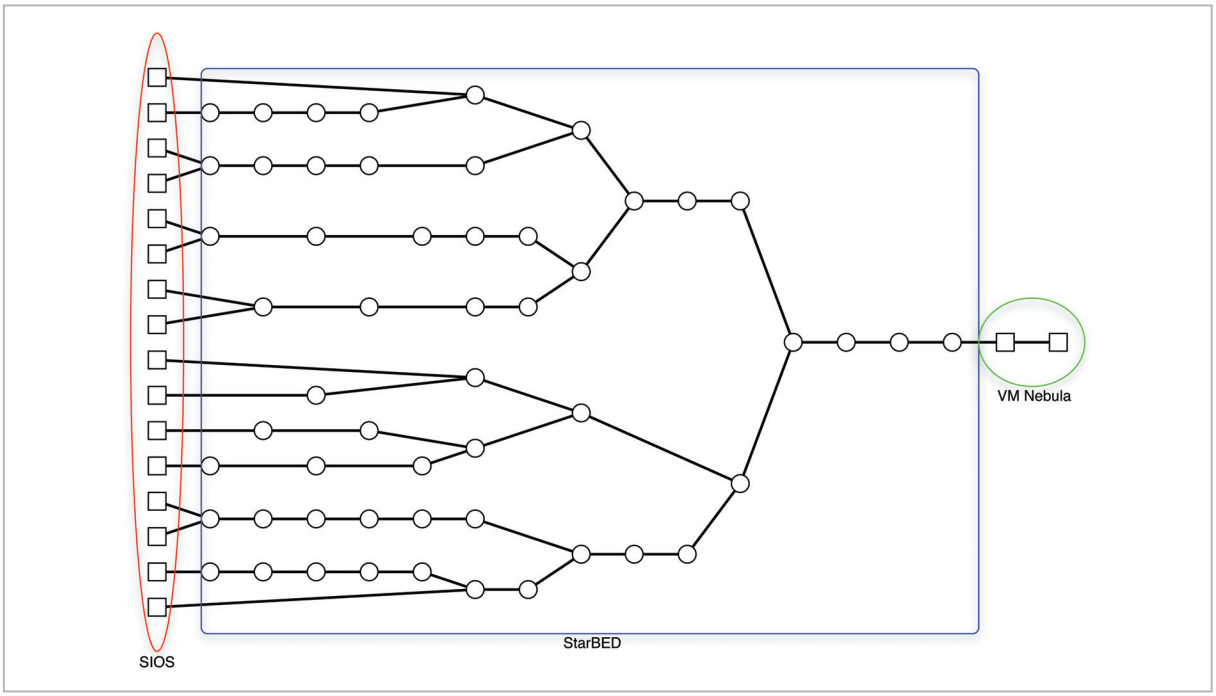

Fig.5 *Schematics of simulation experiment*

to produce the mobile filter function.

On VM Nebula, we constructed a firewall by running the "ipfw" firewall software on a virtual PC and established a web server using the Apache program.

Attacks were made via simultaneous repetition by the 15 attackers of the HTTP request transaction using SIOS's automatic experiment-execution function. The HTTP requests transmitted from the attacker nodes passed through the PC routers on StarBED, passed over the firewall, and reached the web server on VM Nebula.

We confirmed that the target web server became inaccessible immediately after the attacks were launched. When the mobile filter was activated, access to the web server was again enabled.

We should note that the RTT from the attacker nodes to the web server varied widely—between 25 ms and 400 ms—indicating that access was occasionally instable even when the web server was accessible, for example before the attacks and after the activation of the filter.

4 Summary

The following describes our conclusions on the connection of experimental environments based on the results of the experiments conducted as set forth above.

In establishing the connection of experimental environments, various types of information must be distributed through these environments. The most important element in the reproduction of Internet security incidents is considered to be the communication between experimental nodes that simulate the elements of the security incident (attacking host, intermediate host, victim host, etc.). This is because, generally speaking, security incidents on the Internet always rely on communications. An experimental system reproduces a security incident by conducting specific communication procedures and simulating certain symptoms.

When experimental nodes communicate over multiple experimental environments, it is necessary to determine whether the communication bandwidth and actual speeds (throughput, RTT, etc.) are suitable for the communications to be reproduced.

In VM Nebula, all of the experimental nodes used are virtual PC nodes. Therefore, connection of VM Nebula to SIOS or StarBED entails the connection of the VM Nebula virtual PC nodes with the experimental nodes of SIOS or StarBED. As a result, it is important to note connection performance between virtual PC nodes and the SIOS or StarBED experimental nodes.

As shown by the table data, the throughput of UDP_STREAM between virtual PC nodes and the StarBED nodes was either equivalent to VM Nebula internal performance or limited by the performance of the network interfaces of the nodes on the StarBED side. These nodes are categorized based on the network interface, and those corresponding to the network interface required for a given experiment can be selected as needed. Hence, each experimental environment must be designed with the consideration that performance of the virtual PC nodes inside VM Nebula represents the upper limit of connection performance when StarBED is connected.

In actual operation, when multiple virtual PC nodes are running, throughput is expected to be lower than in the results of our experiments. It is therefore important to examine the allocation of elements to be reproduced in the connection with SIOS or StarBED in order to determine the appropriate distribution of these elements.

Although RTT was 1 ms or shorter within VM Nebula, it was approximately 8 ms to 11 ms to or from StarBED. We observed the same RTT value to and from SIOS. Furthermore, when transmission from VM Nebula passed through one PC router on StarBED and reached SIOS, the RTT was approximately 25 ms to 40 ms, thus showing a large variation in speed than those seen within VM Nebula or to and from StarBED.

Accordingly, if we are to obtain detailed results in communication performance testing over multiple environments, it is necessary to take the above difference in speeds into consideration.

5 Conclusions

In this paper we described connection experiments employing VM Nebula and StarBED, systems we developed in the course of previous experiments, as well as simulation experiments using SIOS, StarBED, and VM Nebula. These experiments were carried out to investigate methods of integrating experimental environments in order to reproduce various security incidents. In our report, we also introduced our conclusions given the results of these experiments. Based on our research, we confirmed that a host-to-host communication speed of approximately 100 Mbps is possible without major problems, enabling the execution of experiments reproducing large-scale security incidents.

In the future we plan to verify the effects of the integration of experimental environments by working to reproduce security incidents on a greater scale.

Acknowledgements

Our research expenses are covered by the promotion subsidy for science and technology provided by the Japan Science and Technology Agency. We would like to express our appreciation to those who have enabled us to make use of this valuable research fund.

References

1 Hiroyuki Ohno, Hiroshi Takechi, and Hideki Nagashima, “Internet NO KYOUI NI TAIKOUSIURU ZEI-JAKUSEI database TO KENSHOUSHISUTEMU NO KOUCHIKU”, IPSJ, DSM symposium 2001, Feb. 2001. (in Japanese)

2 Shinsuke Miwa, Osamu Takizawa, and Hiroyuki Ohno, “A Design and Implementation of ‘VM Nebula’”, IEICE, The 2003 Symposium on Cryptography and Information Security (SCIS2003), Jan. 2003.

3 Shinsuke Miwa and Hiroyuki Ohno, “A report on the analysis of Virus and Worm on the VM Nebula”, Internet Conference 2003, Oct. 2003.

4 Toshiyuki Miyachi, Ken-ichi Chinen, and Yoichi Shinoda, “Automatic Configuration and Execution of Internet Experiments on an Actual Node-based Testbed”, Tridentcom2005, Trento, Italy, ISBN 0-7695-2219-X, pp.274-282, Feb, 2005.

5 The Public Netperf Homepage,<URL: http://www.netperf.org/netperf/NetperfPage.html>.

***MIWA Shinsuke**, Ph.D.*

Researcher, Secure Networks Group, Information and Network Systems Department

Networks Security

***OHNO Hiroyuki**, Dr. Sci.*

Group Leader, Secure Networks Group, Information and Network Systems Department

Computer Network, Crisis Management