2-3 A Development of Experimental Environments “SIOS” and “VM Nebula” for Reproducing Internet Security Incidents

MIWA Shinsuke and OHNO Hiroyuki

Security incidents are growing significantly on a daily basis in the Internet world causing considerable damage. To protect against these incidents, some protection mechanisms or softwares has come to be used. Each time a new attacking method or virus or worm appears, it must be analyzed in detail since such protection mechanism or software requires detailed information on that. However, these incidents are sophisticated because enlarged its scale and supervened incidents cause many interactions, make it difficult to acquire such in-depth information.

To analyze sophisticated incidents, we developed experimental environments for reproducing Internet security incidents. In this paper, we report on the development of our reproducing environments called “SIOS” and “VM Nebula”.

Keywords

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

1 Introduction

Modern information and communications systems have developed centering on computers and computer networks. In particular, the Internet has grown into a widespread network connecting general users, private businesses, and even government agencies.

The Internet is a worldwide computer network that serves as a basis for business activities and electronic commerce, and provides ordinary people with an environment in which to exchange and collect information. Many organizations now use the Internet in their communications. Additionally, the Internet is likely to play an essential role in the establishment of so-called “electronic government”.

With the rapid spread of the Internet, information and communications systems have been exposed to growing threats such as corruption or falsification of data, system crashes, denial of service attacks, and theft of personal or confidential information. These threats are now even more pervasive due to the widespread use of always-on broadband connections and increasingly sophisticated attacking techniques.

If the Internet is to fulfill its role as an important element of the social infrastructure, safety and ease of use must be ensured through the elimination of security threats.

Countless security incidents and countermeasures take place on the Internet every day. However, new attack techniques appear one after the other, outsmarting security measures in place; the race seems endless.

Demand is thus high for new countermeasures and technologies that will provide a radical solution.

When devising and developing new countermeasures, we must both verify effectiveness and determine whether these measures have any adverse side effects. To accomplish both of these aims, we have been developing experimental environments for the reproduction of various security incidents[1][2].

In this paper, we will describe the significance and objective of such experimental environments and provide an overview of our developed "SIOS"[1] and "VM Nebula" systems. We will also discuss our efforts to reproduce even larger-scale and more complicated incidents by combining several experimental environments.

[1] SIOS stands for the "Security Intelligent Operation Studio" developed jointly by the Emergency Communications Group of the Communications Research Laboratory (currently NICT) and Yokogawa Electric Corporation. Based on this system, Yokogawa Electric has developed its proprietary "SIOS" product (SIOS is also a registered trademark of Yokogawa Electric).

2 Overview of research on information and communications system security

To outline the technologies we have developed, we will describe the various types of threats to information and communications systems, as well as the types of countermeasures taken against such threats to provide Internet security.

2.1 Threats to information and communications systems

If you connect an information and communications system to the Internet, information is sent or received through an open network; the system is thus always exposed to security threats.

The main types of threats include electronic eavesdropping, tampering, spoofing, illegal access, and denial of service (DoS).

Although these threats are mostly Internet-based, some arise by other means—for example, viruses or worms carried by infected machines, or electronic eavesdropping or DoS through the use of electromagnetic waves.

Since the Internet does not currently provide preventive functions against electronic eavesdropping, tampering, and spoofing, users are required to take independent measures. If these measures are adequate, many threats can be prevented. This paper does not deal with these basic threats, which are mainly addressed using cryptographic and authentication technologies.

On the other hand, it is difficult to prevent illegal access and DoS events that evade security measures through various attack techniques. In the next section, we will briefly describe security measures aimed primarily at illegal access and DoS, as well as the relevant constituent technologies.

2.2 Security measures and constituent technologies

The main illegal access and DoS events are as follows:

- Illegal acquisition and abuse of accounts
- Exploitation of vulnerabilities to cause anomalies
- Forceful waste of resources
- Use of steppingstones to extend the scale of attacks and cause disturbances To address these threats, general users can do the following:
- Determine security policies against individual threats
- Adopt cryptographic and authentication technologies to control access to data
- Install firewalls, IDSs (intrusion detection systems), and antivirus tools to restrict network access
- Eliminate vulnerabilities through OS or software updates and modifications
- Apply security-auditing tools to keep track of security status The above general-user measures are local only. Further measures include:
- Network-wide measures (taken by ISPs)
- Enhancement in security of the Internet itself (carried out by equipment vendors)

IDSs, antivirus tools, and security-auditing tools can respond to known attack methods. These measures are effective only if updated information is provided with each appearance of a new attack technique. OS or software modifications are required whenever new vulnerabilities are detected.

Currently, security measures for information and communications systems are carried out through a two-tiered system: countermeasures are devised and developed against the newest attack method, and then, armed with these countermeasures, vendors, service providers, and general users respond to the attack. In the next section, we will describe the technologies we have developed in light of the current situation.

2.3 Developed technologies

Based on the issues described above, we have divided Internet security technologies into the following categories:

- Technologies to support those devising and developing countermeasures
- Technologies applied by general users

We regard the former category as comprising basic technologies, societal in scope, required to ensure the security of information and communications systems; we view the latter as consisting of countermeasure technologies, including specific countermeasures for various threats (see Fig.1).

In this paper, we will describe the results of our research on the following basic technologies:

- Gathering and sharing of vulnerability information
- Reproduction and analysis of security incidents

In particular, we will focus on experimental environments designed for the reproduction and analysis of security incidents.

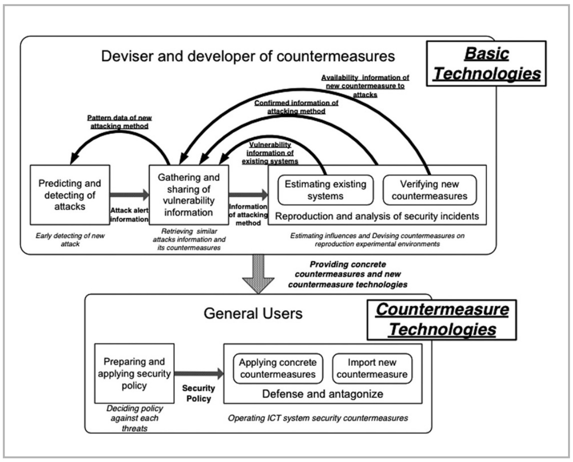

Deviser and developer of countermeasures
Basic Technologies
Availability information of new countermeasure to attacks
Confirmed information of attacking method
Pattern data of new attacking method
Vulnerability information of existing systems
Predicting and detecting of attacks
Attack alert information
Gathering and sharing of vulnerability information
Information of attacking method
Estimating existing systems
Verifying new countermeasures
Reproduction and analysis of security incidents
Early detecting of new attack
Retrieving similar attacks information and its countermeasures
Estimating influences and Devising countermeasures on reproduction experimental environments
Providing concrete countermeasures and new countermeasure technologies
General Users
Countermeasure Technologies
Preparing and applying security policy
Security Policy
Applying concrete countermeasures
Import new countermeasure
Defense and antagonize
Deciding policy against each threats
Operating ICT system security countermeasures

Fig.1 Conceptual drawing of countermeasures and constituent technologies

3 Gathering and sharing of vulnerability information

Today many attackers gain illegal access or disrupt service by exploiting system vulnerabilities, causing what are known as anomalies. Various system vulnerabilities are reported each day, and attack techniques exploiting these vulnerabilities seem to appear without end.

It is therefore essential that we collect system vulnerability information and eliminate vulnerabilities as early as possible. However, a number of problems arise in the collection of this information:

- Difficulty in collecting all information from various sources
- Failure of many sources to disclose information essential in designing countermeasures
- Many sources offer information only in English (not in local languages)

To address these problems, we have been pursuing R&D of a “vulnerability database”[1] to accumulate and organize data on known vulnerabilities. When an attack occurs, this database can be searched for information on similar attacks. By combining this database and the “security incident reproduction equipment” described in Section **5**, we developed a support environment for the formulation of comprehensive security measures, from the assessment of vulnerability data to examination of specific vulnerabilities. We have since filed a patent application[3] with respect to our results.

Information can be entered from various sources in this database. The user can also enter source code that will cause certain vulnerabilities (which normally remains undisclosed to other users) as well as program code for actual attack tools. Using XML (Extensible Markup Language) in data input/output, this database is designed to complement other software applications, providing customizable search methods and means of sharing information among organizations.

Since it is difficult to collect vulnerability information from every source, cooperation among databases is critical. We developed an XML-based method to search across multiple databases of different organizations, and performed a demonstration experiment using the IPA (Information-technology Promotion Agency, Japan) Vulnerability Database and the ICAT Metabase (ICAT Vulnerability Database). Our results verified the feasibility of mutual database searching.

4 Reproduction and analysis of security incidents

When devising countermeasures or developing technologies to protect against attacks such as illegal access and DoS, it is necessary to collect information on the mechanisms and effects of intended attacks, and to verify whether the corresponding security measures are in fact effective.

4.1 Significance and objective of experimental environment for reproduction of security incidents

To collect information on the mechanisms and effects of attacks such as illegal access and DoS, these attacks must be subject to some sort of analysis. To verify whether security measures are effective and whether they pose any adverse side effects, quantitative assessment is required of the effectiveness of these measures and their effects on other systems.

If the code of an attack program is available, the user can perform static analysis of this code to gather data on the relevant attack.

Nevertheless, detailed information about the target system of the attack is required, as problems can also be traced to exploitation of specific vulnerabilities; these problems differ from those presented in ordinary attacks. If several systems are involved, we need to know specifically which system is most likely to be attacked. In these cases, it is difficult to gain a grasp of actual effects using static analysis alone.

Moreover, if the code of an attack program is unavailable, we must estimate the attack mechanisms based on the communications and phenomena generated by that attack. In these cases as well, static analysis is not useful.

If in a given case the information systems are not particularly reliant on the Internet, security measures can be effected directly within the actual systems; when specific problems occur, additional internal system measures can be implemented.

However, we are seeing an increasing number of information systems that rely heavily on the Internet; if user countermeasures have an adverse effect on any of these surrounding systems, major damage may result. Therefore, it is necessary to verify beforehand whether the intended measures are effective and whether they will have any effects on other systems. In this verification process, it is important to understand the interaction among systems, which is also difficult to grasp using static analysis alone.

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. In this case, the experimental environment should be able to emulate the entire system, reproduce attacks, and simulate countermeasures. Accordingly, we have been working on the development of experimental environments to reproduce security incidents, to establish a basis for the reproduction and analysis of security incidents and verification of countermeasure technologies.

4.2 Requirements of experimental environments to reproduce security incidents

There are six main types of incident reproduction/simulation experimental environments involving Internet security:

- Traffic generator

  Simulates communications only by generating specific communications related to attacks

- Network simulator

  Creates simulation models by representing network components as abstract nodes
- Network emulator

  A network simulator with added communications functions
- Experimental environment consisting of PC and router emulators

  Uses PC and router emulators to emulate hosts and gateways, main network components
- Experimental environment of actual PCs and routers

  Uses real machines to simulate main network components
- Experimental environment on the actual Internet

  Uses the Internet to conduct testing

These experimental environments differ considerably from one another in the ability to reproduce threats, varying significantly in terms of accuracy, for example. Generally speaking, more accurate experimental environments are less scalable and more difficult to operate, while less accurate experimental environments (using more abstract methods) are more scalable and easier to operate.

What types of experimental environments are needed to analyze security incidents and to devise countermeasures?

The main types of security incidents seen in recent years consist of exploitations of vulnerabilities and DDoS (distributed denial of service) incidents, which feature the following:

- A large number of steppingstones to increase the scale of attacks
- Complicated attack mechanisms such as spoofing
- Exploitation of vulnerabilities unique to certain OSs or programs

Many security incidents are caused by vulnerabilities unique to certain OSs or software applications. To isolate the cause of an incident, you need to have an environment that can reproduce specific vulnerabilities. To measure the capacity, effectiveness, and effects of a particular countermeasure technology, you must be able to implement the technology and perform a conformance test.

With the increase in the range of effects of security incidents in recent years, more countermeasure technologies are being designed for extensive use. To analyze and verify these technologies, a large-scale experimental environment is required.

5 Experimental environments for reproduction of security incidents

Based on the requirements described above, we have developed:

- An “illegal packet simulator” that can simulate large volumes of illegal communications[4]
- “Security incident reproduction equipment (SIOS: Security Intelligent Operation Studio)” that can reproduce the mechanisms of complicated attacks on real machines[1]
- The “VM Nebula”, which uses virtual PCs to simulate vulnerabilities unique to certain OSs or programs

In this section, we will briefly describe each of the above.

5.1 Illegal packet simulator

Fig.2 Illegal packet simulator

The illegal packet simulator (Fig.2) is a traffic generator that can simulate large volumes of illegal packets caused by DDoS attacks or worms. This simulator is designed for use in penetration tests designed to collect data only on the attacks themselves (data on attacker behavior is unnecessary in this case).

We developed this simulator based on the traffic generator. This simulator can generate packet patterns based on predetermined parameters such as volume and number of occurrences.

We also developed a descriptive language for the creation of packet patterns to allow this simulator to generate all types of illegal packets. As a result, this simulator can send 480,000 packets per second, and we verified that the generated volume of illegal packets can fill 98.7% of a 1-Gbps Ethernet line.

5.2 Security incident reproduction equipment (SIOS: Security Intelligent Operation Studio)

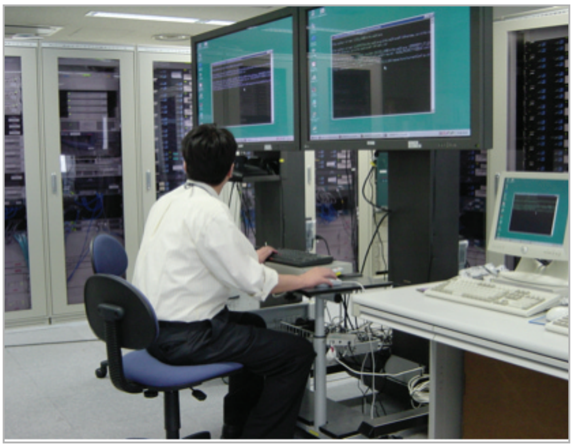


Fig.3 Security incident reproduction equipment (SIOS: Security Intelligent Operation Studio)

The security incident reproduction equipment (Fig.3) provides an experimental environment in which real machines can be used to reproduce attacks accurately. This experimental environment consists of: an attack reproduction segment with 100 PCs; an Internet segment with controllable bandwidth; and a victim segment consisting of DNS/SMTP/Web servers in addition to firewalls and IDSs.

This experimental environment is able to reproduce complicated attacks, including spoofing, using actual attack tools within the reproduction segment, and to assess the effects of these attacks and vulnerabilities within the victim segment. We have filed a patent application[5] with respect to the developed system.

Generally, it is difficult to manage experiments or perform measurement in a large experimental environment that uses real machines. SIOS, however, makes it easy to carry out detailed management of experiments using a feature that supports operational condition settings and automation. We have filed a patent application[6] covering this feature as well.

We developed the equipment to form a support environment for the formulation of a comprehensive range of security measures. In conjunction with a “vulnerability database”, the equipment can be used to load attack tools or to assess vulnerability. It is also able to examine the vulnerabilities of various devices by connecting these devices to the victim segment. Moreover, we performed a combination study with an early-warning system used in the prediction and detection of attacks, and developed consistent security measures for experimental environments[7] based on the early detection of attacks.

Furthermore, we developed a functioning environment in which to practice responses to actual security incidents generated by the equipment. Using this feature, we tested the practice responses of the NIRT (National Incident Response Team, Cabinet Secretariat).

5.3 VM Nebula

VM Nebula (Fig.4) uses a PC emulator to run ordinary OSs and applications on virtual PCs. It can accurately emulate vulnerabilities unique to certain OSs or applications. This environment consists of several emulation servers, a multilayer switch linking these servers, and a library server that manages the system configuration.

With a real-machine environment, modifying the physical configuration is not easy. On the other hand, modification is required in the configuration of the experimental environment according to the purpose of the experiment, such as a study of vulnerability or an analysis of attack behavior. With a simulator, it is easy to modify the configuration, but it is difficult to reproduce vulnerabilities found in actual OSs or applications.

VM Nebula uses virtual PCs to emulate the attacker's PC and the victim's server, and uses a VLAN to connect them, bypassing to need to modify the physical network configuration. By saving the configuration to the library server, you can readily load it to repeat an experiment or to make necessary modifications. In this way, VM Nebula allows for easy switching among different experiments. This feature also enables convenient repetition of experiments that have a disruptive influence on the experimental environment. In particular, we have confirmed the effectiveness of VM Nebula in analyzing viruses and worms[8].

Although the illegal packet simulator, security incident reproduction equipment, and VM Nebula feature different characteristics, all are designed to reproduce and analyze security incidents to assist in the execution of countermeasures. We studied combinations of these environments[9] and proposed a specific method for combined operation. We will discuss this combined operation in detail in Section **7**.

Fig.4 *VM Nebula*

6 Application examples of incident reproduction experimental environments

We applied the incident reproduction experimental environments described above to the development of countermeasure technologies and the analysis of viruses and worms. In this section, we will describe two application examples.

6.1 Vulnerability resistant cluster

To implement security measures against illegal access or other attacks, protective or countermeasure technologies are required, such as firewalls or IDSs.

In most cases, however, these type of technologies can protect systems against known attacks, but not against unknown attacks. Some types of servers, such as web servers, must be open to access by anyone in order to provide the relevant services. It is difficult to protect these servers.

To address this problem, we developed a "vulnerability resistant cluster"[10] to protect against unknown attacks.

To combat exploitation of vulnerabilities, a prominent type of attack these days, this cluster automatically switches OSs or service software upon detection of an attack.

Since vulnerabilities are unique to certain OSs or software programs, a given exploitation will be harmful to a limited number of OSs or software programs featuring the relevant vulnerability. Based on this, the vulnerability resistant cluster neutralizes exploitation of the vulnerability.

With the aim of using this cluster technology in servers available to the public, we developed a "vulnerability resistant server system"[11] using virtual machine technology.

When developing the vulnerability resistant cluster, we needed to make sure that this cluster would function effectively with an actual unknown attack on the system. Accordingly, we set up an attack host and a vulnerability resistant cluster in the experimental environment, VM Nebula, and simulated an unknown attack to see how the cluster would react.

6.2 Analysis of MS.Blaster worm

When analyzing a virus or worm, it is necessary to check routes and means of infection, targets, traces, effects, and functions after the infection. To assess these items, static analysis is applied to the virus or worm's program code. However, if the virus or worm features complex behavior, it is difficult to collect a sufficient amount of data using static analysis. Therefore, it is often necessary to conduct dynamic analysis as well.

When conducting dynamic analysis, it is necessary to cause an actual infection. Therefore, these experiments must be conducted within an isolated environment to prevent leakage of the virus or worm. To reproduce a specific vulnerability, the experimental environment needs to reproduce an attack as well in simulation as with actual machines. If a virus or worm sample is allowed to cause an actual infection, PCs or other nodes within the experimental environment will be infected. However, it is difficult to conduct a full analysis or verification by causing an infection only once. Therefore, the experimental environment must be able to return to the disinfected state so that this procedure can be repeated many times.

In the case of a real-machine environment, it takes a great deal of time and effort to reinstall OSs and software programs each time the machines are infected.

With a licensed VM Nebula system, you can save an experimental environment in an uninfected state in a library, and then reload it as necessary to restart the experiment without delay after the environment is infected.

Using this feature, we conducted the otherwise onerous dynamic analysis of the MS.Blaster worm (see Fig.5)[8] and verified the effectiveness of this environment.

7 Limitations of the single experimental environment and combined operation of several experimental environments

Up to this point, we have described reproduction experimental environments for use in the analysis of the mechanisms of security incidents and the evaluation of the effectiveness or adverse effects of countermeasure technologies. 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.

In this section, we will describe the limitations of single experimental environments and combined operation of several experimental environments.

7.1 Limitations of single experimental environments

Various experimental environments have a range of advantages and disadvantages. The size and configuration of an environment are limited depending on the implementation. Generally, an experimental environment is created to address incidents that occur within a certain period. Therefore, it is possible to specify the size and intended incident according to conditions set at the design stage. However, since actual conditions change constantly, the configuration of the experimental environment must be modified to address such changes.

In the case of a network simulator or emulator, it is possible to increase the size of the environment to a certain extent only by increasing the number of internal nodes; this method does not increase the cost of the simulation. On the other hand, in the case of an experimental environment made up of real machines, an increase in environment size usually means the addition of physical nodes, at considerable added cost.

This real-machine experimental environment can usually adapt to changes in intended incidents through the OS adjustment or modification in the node software. On the other hand, to change the settings of intended incidents in a simulator or emulator, the software must be reinstalled from scratch, and some incidents remain difficult to reproduce.

In short, it is difficult for a single experimental environment to adapt to changes in the

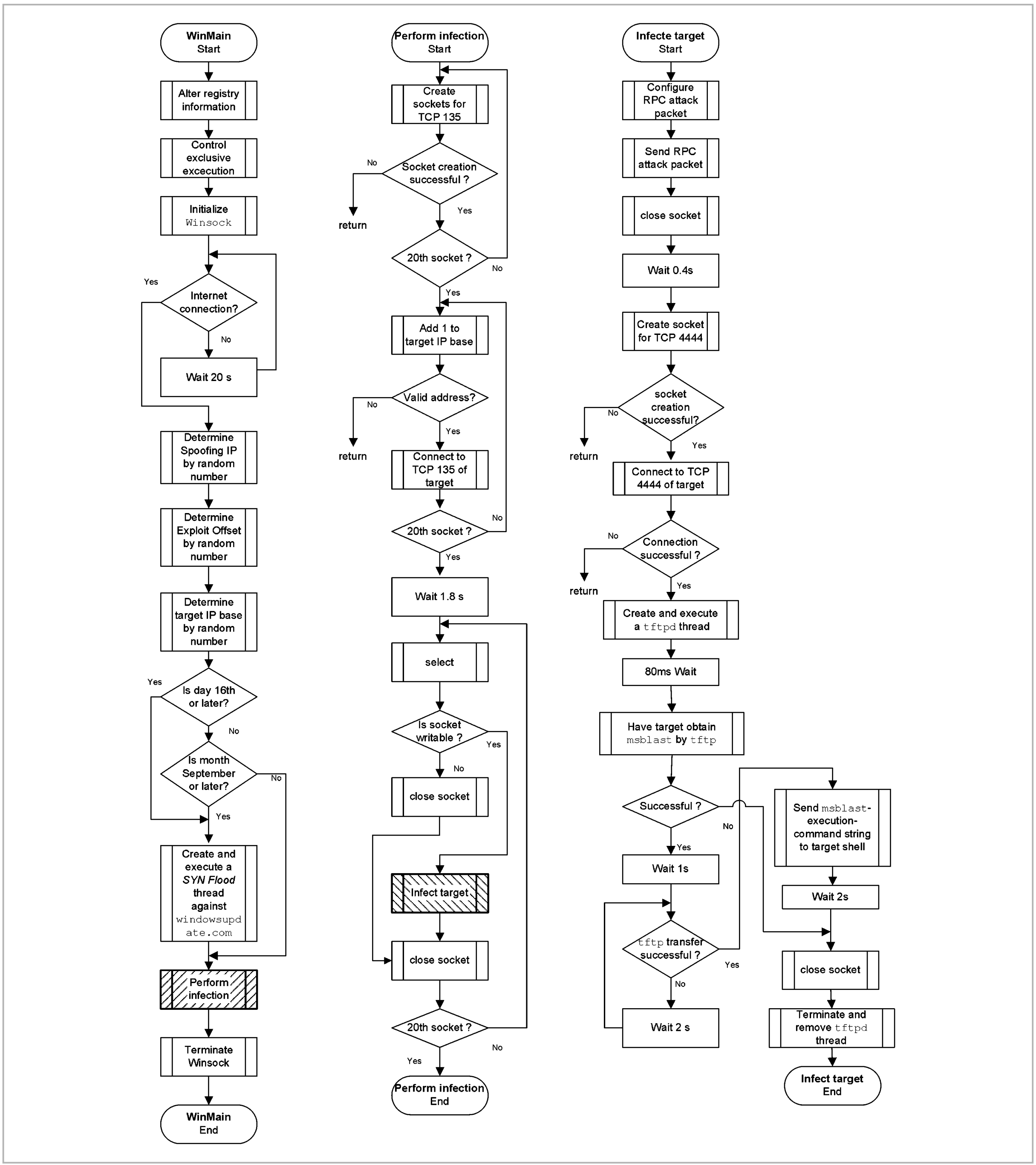

WinMain Start
Alter registry information
Control exclusive excecution
Initialize Winsock
Internet connection?
Yes
No
Wait 20 s
Determine Spoofing IP by random number
Determine Exploit Offset by random number
Determine target IP base by random number
Is day 16th or later?
Yes
No
Is month September or later?
No
Yes
Create and execute a SYN Flood thread against windowsupdate.com
Perform infection
Terminate Winsock
WinMain End
Perform infection Start
Create sockets for TCP 135
Socket creation successful ?
No
return
Yes
20th socket ?
No
Yes
Add 1 to target IP base
Valid address?
No
return
Yes
Connect to TCP 135 of target
20th socket ?
No
Yes
Wait 1.8 s
select
Is socket writable ?
Yes
No
close socket
Infect target
close socket
20th socket ?
No
Yes
Perform infection End
Infecte target Start
Configure RPC attack packet
Send RPC attack packet
close socket
Wait 0.4s
Create socket for TCP 4444
socket creation successful?
No
return
Yes
Connect to TCP 4444 of target
Connection successful ?
No
return
Yes
Create and execute a tftpd thread
80ms Wait
Have target obtain msblast by tftp
Successful ?
No
Yes
Wait 1s
tftp transfer successful ?
Yes
No
Wait 2 s
Send msblast-execution-command string to target shell
Wait 2s
close socket
Terminate and remove tftpd thread
Infect target End

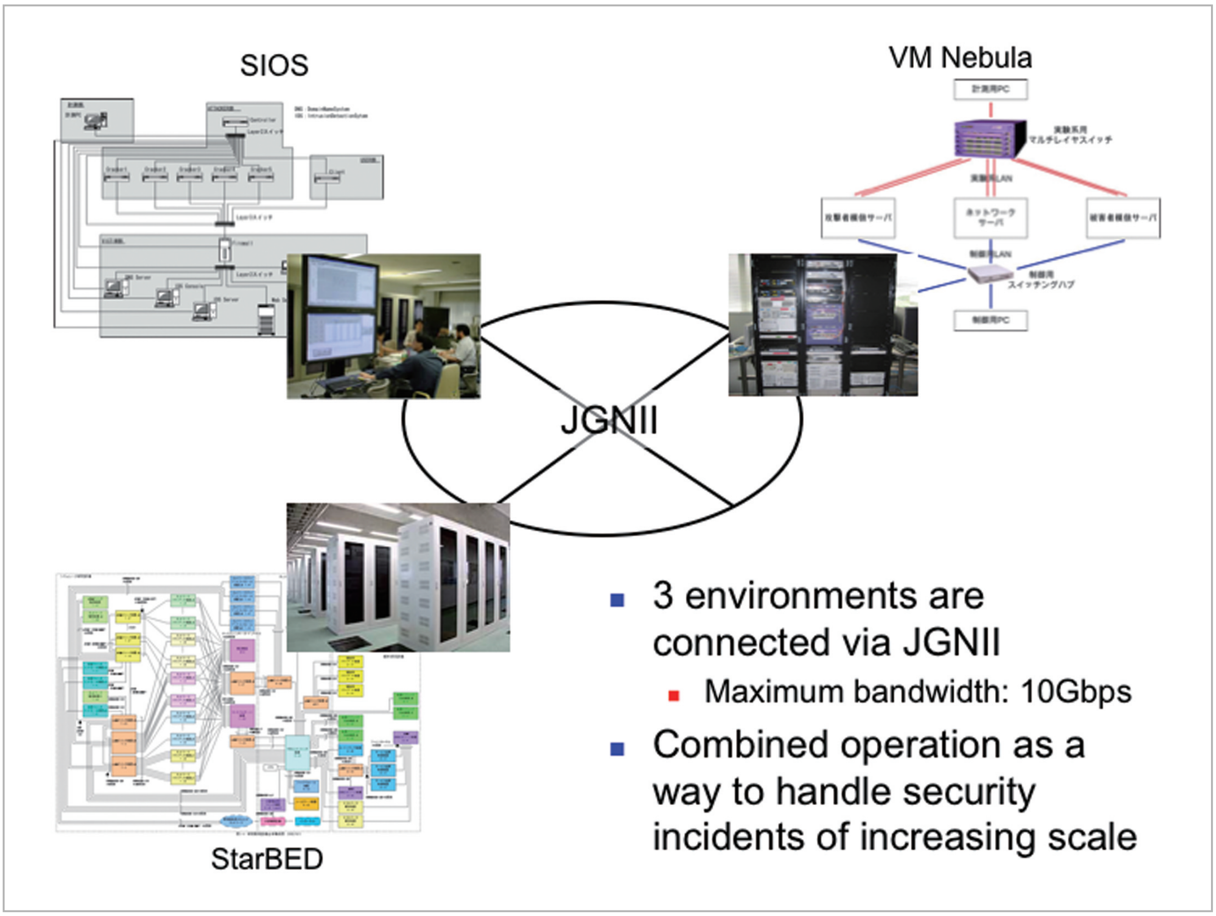
Fig.5 *Behavior of MS.Blaster worm (analysis results)*

conditions of security incidents. Considerable costs are incurred when accommodating such changes.

7.2 Combined operation of several experimental environments

As described in the preceding section, a single experimental environment has a limited capacity for reproduction or simulation due to its design. However, to change the basic conditions of a security incident freely, this limited capacity will not suffice. If money were not an object, you could make all of the required modifications for the desired conditions, but in reality costs must always be controlled.

In this section, we will discuss the combined operation of existing experimental environments at minimum cost as a way to handle



- 3 environments are connected via JGNII
  - Maximum bandwidth: 10Gbps
- Combined operation as a way to handle security incidents of increasing scale

Fig.6 *Combination of three experimental environments*

security incidents of increasing scale and complexity.

For example, through the combination of experimental environments, it will become possible to:

- Increase the number of nodes (real machines). For example, you can combine 100 nodes of one environment and 500 nodes of another to create an environment with 600 nodes; or
- Link a network emulator to machines to enhance both reproduction performance and scalability.

Nevertheless, we cannot operate several experimental environments in combination simply by linking them together. To conduct effective Internet security experiments, these environments must be able to communicate and exchange measured/recorded data. They also need to operate in sync, not just separately, on an event- or time-basis. In particular, inter-system communication responses must occur in the correct order.

Many issues therefore remain, such as how to connect and combine several experimental environments and how to address differences among these environments[12].

7.3 Trial operation of several experimental environments in combination

To study the combination of experimental environments, we are planning to test the combined operation of the three experimental environments[13] we have developed to date (see Fig.6):

- Security incident reproduction equipment (SIOS, see **5.2**)
- VM Nebula (see **5.3**)
- StarBED^2

These three experimental environments are geographically distant from one another. We had to link them physically and provide a remote control function in order for these environments to interoperate.

As for the physical connection, there were three tentative plans: use of a dedicated line only, combination of wide-area LAN (shared within NICT) and a dedicated line, and JGN2. We selected JGN2 for the following reasons:

- JGN2 provides broadband network connection (at 10 Gbps).
- JGN2 is a dedicated test network and is isolated from the Internet.
- All three experimental environments are located near a JGN2 access point, enabling low-cost interconnection.

We will use JGN2's multipoint simultaneous connection service to connect the three points via Ethernet and to switch among several VLAN settings as necessary[3]. Transmission speed between SIOS (in Koganei, Tokyo) and StarBED (in Nomi, Ishikawa Pref.) is 10 Gbps, and that between SIOS and VM Nebula (in Kobe, Hyogo Pref.) is 1 Gbps.

We are planning to use VLANs for the purposes listed below. We have prepared four VLANs (one of which is a redundant network) for normal operation.

- Operation and control of the experimental environment, and data measurement
- Mutual remote control
- Communications between nodes used in experiments

Since each experimental environment uses a private IP address space with its own rules, it is necessary to make arrangements concerning the logical connections (such as allocation of IP addresses, whether to use routing, and routing methods). We must also take topology into consideration (e.g., which segments are to be connected among the experimental environments, and how).

As a means of remote control, we have decided to use existing "KVM over IP" equipment at each experimental environment for the time being; we will remotely control keyboards and mice with on-screen operations from each environment. However, it is difficult to operate and manage the equipment in this manner; we are therefore planning to prepare a unified interface.

All of these experimental environments operate on a real-time basis with an actual OS and software configuration. Although we are planning to use NTP for time synchronization, other means of time synchronization are available for consideration. Other than this, we will not adopt any functions to combine these environments.

Combined operation is still in the preparatory stage as of this writing. We intend to report on this operation after the connection is established.

8 Conclusions

In this paper, we have discussed experimental environments that will provide a basis for the development of technologies to address security threats on the Internet. Specifically, we have described our R&D activities relating to equipment or experimental environments for the reproduction and analysis of security incidents, and the combined operation of several experimental environments.

Going forward, we will work on the creation of experimental environments that can handle large-scale, complex incidents, so that we can devise radical and comprehensive solutions to address a wide variety of Internet environments.

[2] StarBED is the nickname for a simulator facility at NICT's Hokuriku IT Open Laboratory (in Nomi, Ishikawa Pref.). StarBED is a general-purpose cluster consisting of 512 PCs for use in network-related experiments.

[3] Such settings are normally not available, but are provided particularly for this purpose. VLAN settings will be switched manually.

Reference

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.

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 Yutaka Yokochi, Izumi Yamamoto, Hiroshi Takechi, Hidemi Nagashima, and Hiroyuki Ohno, "Vulnerability examination system", Japanese patent, Published patent application No.2002-229946, Jan. 2001.

4 Shinsuke Miwa, Osamu Takizawa, and Hiroyuki Ohno, "A Concept and Design of DDoS attack generator", IPSJ, DPS111-22&CSEC20-22, Feb. 2003.

5 Hiroyuki Ohno, Satoshi Iokura, Hidemi Nagashima, and Hiroshi Takechi, "Vulnerability examination system for compter system", Japanese patent, Published patent application No.2002-229945, Jan. 2001.

6 Hiroyuki Ohno, Hiroshi Takechi, Hidemi Nagashima, Hiroki Yanagibashi, and Kazuya Kubo, "Operating condition setting method in distributed environment and system using this", Japanese patent, Published patent application No.2002-229877, Jan. 2001.

7 Shinsuke Miwa and Hiroyuki Ohno, "Design and Concept of integrated Internet Security Environment", IEICE, The 2002 Symposium on Cryptography and Information Security (SCIS2002), Jan. 2002.

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

9 Shinsuke Miwa, "A study of an integration of simulating environments for the Internet security incidents", IPSJ, DPS workshop 2003, Dec. 2003.

10 Shinsuke Miwa, "Vulnerabilities Resistant Cluster" - HA Clustering against cyber attacks using vulnerability exploits, JSSST, WIT2003, Nov. 2003.

11 Shinsuke Miwa, "A design issue of the "Vulnerabilities Resistant" server system with Virtual Machine technologies", JSSST, WIT2001, Sep. 2001.

12 Shinsuke Miwa and Hiroyuki Ohno, "A study of an integration method of simulating environments for the Internet security incidents", IPSJ, DPS workshop 2004, Dec.2004.

13 Shinsuke Miwa, Toshiyuki Miyachi, Hiroyuki Ohno, and Yoichi Shinoda, "A study of an experimental environment integration method for reproducing Internet security incidents", IEICE, The 2005 Symposium on Cryptography and Information Security (SCIS2005), Jan.2

***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