15th Annual Computer Security Incident Handling (FIRST) Conference

Ottawa, Canada
June 22–27, 2003

Summarized by Anne Bennett

The Forum of Incident Response and Security Teams (FIRST) is a global organization whose aim is to facilitate the sharing of security-related information and to foster cooperation in the effective prevention and detection of, and recovery from, computer security incidents. It holds several technical colloquia each year open to members only, and one annual conference which is open to all.

TUTORIALS

Network Forensics

E. Larry Lidz, University of Chicago

Looking at problems from the network: Often, we don't have access to the machine we suspect of being involved in a compromise (the machine is physically inaccessible, it belongs to a student, etc.). Network audit logs can save the day: If we are able to go back and show where a problem came from, we can quickly resolve the problem. Also, if it is necessary to turn over evidence to authorities, it can be legally easy to turn over network audit logs, which tend not to contain confidential information, whereas it can be really tricky to turn over the hard drives of compromised machines (because confidential information must be protected).

The speaker gave a crash course on TCP/IP, connection establishment and termination, IP addresses and network masks, and well-known ports.

Why use network audit logs if we already have one or more of an IDS (Intrusion Detection System), the ability to do system forensics later, the ability to sniff traffic as needed, or firewall logs?

- IDS: IDS logs only known suspicious traffic, but we often need complete logs. Also, some kinds of suspicious packets cannot be logged: Out-of-sequence FINs can "turn off" IDS monitoring for a connection, and too many fragments can overwhelm an IDS.
- System forensics: A compromised system has often had its data altered, attempts to fix the problem may have destroyed data, and the intruder may not have written anything to disk.
- Sniffers: Often it is too late to turn on the sniffer – the problem has already happened. Also, because the entire packet is logged instead of just the header info, disk consumption is massive and there are more serious privacy issues.
- Firewalls: Usually they do not contain as much information as audit logs (except in the special case of application firewalls).

All of the above have their place, but it is still useful to have network audit logs to help investigate break-ins, to determine how our network is being used for both legitimate and illegitimate purposes, and to get a picture of "normal" use of the network. Also, the network audit logs are (or should be!) on a trusted, hardened system, so their information is quite reliable.

Well-known audit log systems:

- Cisco NetFlow logs straight from Cisco (and some other vendors') routers and switches (can impact performance). These logs are easy to search.
- Argus logs from the spanning port of a switch and is able to log packet contents if desired. Argus is a free product, and there is a more DoS-resistant and featureful related commercial product named Gargoyle.

Active probing (port scanning), though useful during investigations, may tip off an intruder that you have noticed them. Make sure you have the authority to scan any machine you want to probe.

Some useful tools are:

- nmap: finds open ports, O/S fingerprinting.
- nc: text communication with tcp port.
- pnscan: a massively parallel scanner, can grab banners, even pass text to the port, but can sometimes miss hosts because of timeout problems.
- rpcinfo: identifies RPC services running.
- smbclient: NetBIOS info, comes with Samba.
- dcetest: like rpcinfo but for DCE services.
- ibnet: can craft specific packets.

It is extremely important to keep system logs in a trusted place (central log server) and watch them for unusual activity.

We were shown examples of how NetFlow and Argus are used to investigate incidents. For example, a bunch of Solaris hosts were reported to be compromised. The investigators picked a relatively "quiet" one (fewer logs to look through) and found out that something had scanned it for port 111 and connected to ttdbserver. Then another host had connected to port 1524 (which was later determined to be a back-door port) and rsh'd to a machine on the same network as the original "scanner" (this was interpreted to mean that the victim was downloading a rootkit); then IRC traffic started up.

This analysis suggests that to find additional compromised machines, there are a few things that can be looked for: successful connections to port 1524 locally, local machines rshing to the same offsite machine, and, possibly, IRC traffic

(unless there is lots of legitimate IRC traffic). Ideally, you should cross-correlate multiple sources for better reliability, and, of course, hand-verify hosts likely to have been compromised.

Hints on what to look for in network logs when investigating an incident:

- When was the first traffic to a back door (first successful connection to a port which previously had no traffic)?
- When did IRC traffic start?
- Was there a sudden increase in traffic on a given machine?
- Did unexpected traffic start shortly after a connection to a service running on the machine?
- Once a back-door port has been found, what machines connected to it?

These techniques proved useful for Slammer Worm cases (where the symptoms were that the external net died and several internal routers crashed; Cisco reported a network-wide DDoS) and in cases of a specific compromise. The initial Slammer analysis and identification of an initial set of Slammer-compromised machines was done in 15 minutes!

Note: There are now tools (e.g., softflow) to send flow logs off from UNIX machines which are acting as routers.

Secure Coding: Principles and Practices

Kenneth R. van Wyk, Tekmark Technology Risk Management, USA; Mark G. Graff

The intention of the authors of the O'Reilly book of the same name was not to give tons of source code examples but, rather, to give programmers the ability to think clearly about secure coding. When we build bridges, we don't make trucks drive over it, see if it collapses, then if it does, move on to "bridge v.1.1." We need a similar perspective for the construction of software.

Van Wyk strongly emphasized a mindset that needs to be adopted to have any chance of creating reasonably secure software. Try to look at your software from the point of view of an attacker: how can it be subverted? With that in mind, it is possible to apply a set of architectural principles and make sure that the proposed design respects those principles, such as: restrict privileges granted, assign responsibility to individuals (make sure that actions can be traced back to their originators), log sufficiently well that events can be reconstructed, and fail safely.

We were given examples of risk-assessment questions and broad categories of risk mitigation strategies, including avoiding the risk by removing the cause or the consequence, limiting the risk by detecting and responding to problems, transferring the risk to another party (e.g., by buying insurance), and assuming the risk (being prepared to deal with the consequences of a problem).

When maintaining or modifying existing software, be aware of the design intent and the security model of the original, otherwise you risk introducing problems you have no idea about.

Near the end of the presentation, we were given some specific coding tips. The fairly low emphasis on this issue in the presentation maps well onto the amount of time that should be devoted to the implementation of a project, versus planning and design, which should receive the lion's share of time. Most of those coding tips ought to be well-known by now!

Finally, don't neglect the environment in which the code will be used: Secure the OS and network, monitor logs, keep up to date with patches, and so on. And, of course, don't forget to test every component and test at every stage: Your environment changes. Automate your testing.

Creating a National Alerting and Reporting Service

Nienke van den Berg, GOVCERT.NL, the Netherlands; Jeffrey Carpenter, CERT/CC, Carnegie Mellon University, USA; Graham Ingram, AusCERT, Australia

The Internet is no longer just an academic and research network, but has become part of a national social and economic infrastructure. Governments are starting to look to national CSIRTs to help keep the Net viable as such an infrastructure. Entering a "national picture" has raised legal, process, alerting (info provided to the public), and reporting (info received from the public) issues for existing (and new) CSIRTs.

In deciding when to issue an alert, you should evaluate the particular vulnerability or event based on its technical impact (a.k.a. objective impact – the probability of technical and economic damage) and on its social impact (a.k.a. subjective impact – the perception of safety, influenced by the likelihood of media exposure).

How to provide alerts: Web (need a good content management system), email (lists can get quite large), mass media, and SMS.

Not all of these are used for each alert: There is a decision matrix that maps technical and social impacts onto a "media mix" (set of channels through which the alert is sent). A communications advisor is used to ensure that alerts are "not too technical" when they are aimed at the general public.

Email alerts are PGP-signed, of course, but since the general public may not be in a position to check the signature, it helps to have a very well–publicized Web site address to reduce the chances that an impostor could hijack (forge) the distribution channel.

Here are some questions to ask in deciding what is worthy of advisories: What is the current threat posed by this vulnerability? How bad would it be if the vulnerability were publicly known? if the vulnerability were publicly known and being exploited? How bad is the incident already? How much worse can it get?

CERT/CC has developed formal metrics to rank vulnerabilities according to the severity of the impact on a "typical" site; these metrics involve questions with numerical answers by which a weighted total is computed. This metric is used to help decide whether the problem justifies an email advisory or whether it is sufficient to post the info to the Web site. The metric is also used to decide which issues to work on first. The questions involve the ease of exploitation of the vulnerability, how widely known the vulnerability is, the risk to the Internet infrastructure, and how many systems would be affected and the severity of this impact.

CERT/CC has also developed formal metrics to rank incidents (as opposed to vulnerabilities). With respect to incident activity relating to a particular vulnerability, CERT/CC differentiates between current activity and potential additional activity, not counting what has happened so far (exploitation of a vulnerability), in order to determine the goals of any action.

CERT/CC also adds a "uniqueness factor" so that they don't issue multiple advisories or incident notes on the same subject, even if related-incident activity continues.

To evaluate current impact, questions are asked about how many people and/or machines are affected, at how many unique sites, what the importance of the systems affected is, what the impact during and after the actual activity is, and how complicated the attack method is.

To evaluate potential (additional) impact, questions involve how many people and/or machines are affected by the vulnerability, the importance of the systems, how rapidly the problem is spreading, and how complicated the attack method is (i.e., how many people could write an exploit, therefore how likely the exploit is to appear soon).

To set up an alerting service you need money; an operational CSIRT (center of operations); systems for Web service, Web content management, email list management, and project and office management; and technical, communication, and legal expertise. Legal issues must be addressed (general terms and conditions of service, privacy policy and disclaimers, contracts and service level agreements), PR must be handled, and internal processes must be clear.

The purpose of a reporting service (e.g., AusCERT) is to collect, process, and analyze computer security incident reports and share sanitized aggregate reporting with an appropriate audience. This should provide meaningful intelligence about trends and modus operandi with respect to network attack activity.

The goals of the service are to promote the use of mitigation strategies, raise awareness of computer security issues, keep people up-to-date with threat activity and trends, provide information about attack data which they would otherwise not be able to obtain, and provide value-added analysis of this data.

Any reporting service must be able to protect the data of the reporters (and reassure them that this will indeed be the case). However, a reporting system as a black hole is not a good idea: People are motivated to provide data by having access to sanitized and/or aggregated data and statistics.

It is important to have a clear workflow for the reporting process, not least because it forces you to make sure that your stated goals are in fact being addressed in your processes. Also, privacy considerations dictate that you must have guidelines about what information is communicated to whom and under what conditions. Finally, the process makes clear what resources are needed to do the work.

AusCERT described their complex reporting service system. First of all, it was necessary to collect incident data online; the alternative of a call center staffed by 40 or so people was simply not viable. In terms of security, the last thing a CSIRT wants is to have the press report that the CSIRT suffered an intrusion! It is equally important to keep data segregated, so that one reporter does not have access to data submitted by another reporter. The automated response system must automatically do triage on the incoming data: If something is reported for the first time, it probably requires human attention. The system must assign "threads" to incidents so that correspondence and actions on a particular incident can be correlated.

The types of requirements that must be met in order to set up a reporting service are the same as for an alerting service, but the actual contents are different. Most CSIRTs need to use a Web form to collect data in order to automate the process, including the initial analysis of incoming reports. The system must scale very well. A worm might result in thousands of reports from the general public, for example, but if there is a new and dangerous exploit nestled in among these thousands, the system must quickly pick it out for human attention. It must also be possible to quickly change the Web form to respond to emerging issues; for example, a new type of incident might require the collection of a new category of data.

How to handle anonymous reports? AusCERT accepts and stores anonymous

reports, but takes no action unless they can later be correlated with reports from validated sources. Don't forget to try to collect consent from reporters with respect to sharing confidential information with specific bodies when needed. Arrange to refuse or otherwise deal with certain types of reports that are not computer-related or where liability would be an issue – reports of pure criminal activity such as murders, for example, are not wanted in a computer security incident reporting scheme. Reports of cybercrime accompanied by a refusal to release information to law enforcement authorities should also be discouraged or refused.

Alerting and reporting services tips and tricks. Share knowledge and expertise with other CSIRTs; integrate the alerting and reporting activities with the processes of the CSIRT; do alerting first to build credibility for later report collection; keep close contact with target groups to improve the quality of the alerts; start a newsletter service for people who are not specifically interested in alerts; and establish good national press contacts in case escalation is needed.

Introduction to Advisories

Andrew Cormack, UKERNA, UK

1. Why do vulnerabilities happen?

- Laws of nature: Because computer networks are complex systems, they will certainly contain errors, some of which will have security implications.
- Customer demands: People ask for computers which are "easy" to use; rarely do they demand computers which are "safe" to use. Therefore, vendors sell systems with everything turned on. When Sun tried the opposite, they received many "non-working" returns! And while users will turn on what they need, they will rarely bother to turn off services they don't need.
- Vendor pressures: Vendors are under economic pressure to ship new features fast, with the result that testing is incomplete. Testing for security (the absence of unintended functionality) is harder than testing for intended functionality.

2. Sources of information about vulnerabilities:

- Incident reports are clearly a reliable indication that there is a problem! However, the information obtained may have been obscured by the attackers and may be hard to interpret.
- Full-disclosure communities (e.g., BugTraq): Often the information is up to date but the quality is variable, and there is more emphasis on problems than on solutions.
- Crackers (via published tools): The information is very current, but malware has to be handled with extreme care, since it may contain additional attacks aside from the attack it claims to be performing. It can be difficult to extract good information from the tools; reverse engineering is required.
- Vendors: Information can be of very good quality, but vendors can be very slow, and, because of competing motives, the information may be incomplete or may understate the impact of the problem.
- Commercial services such as antivirus vendors, ISS, etc.: The quality of the information is generally high, and their advisories usually come out before the vendors' advisories. However, there may be restrictions on distribution, and, again, competing motives may affect the information, for example, by overstating the impact in order to sell their services.
- Other CSIRTs have similar motivations as us and are generally trustworthy. But there may be restrictions on distribution of the information. In addition, CSIRTs may be slow, depending on their policies and on the resources available to them.

Clearly, not all information is equal, so it helps to use multiple sources to ensure speed, reliability, and completeness. You should verify this information by correlating the information from independent sources, testing for yourself, and using the trustworthiness of the source as a criterion.

3. CSIRT tasks with respect to vulnerabilities:

- Planning: Plan in advance how to use information for the benefit of your constituency and minimize harm; there's no point in sending out an advisory that says, "There's a problem but there's nothing you can do."
- Distribution: Simply pass on existing information, perhaps translating to a local language or sending only information that is relevant to a particular constituency.
- Interpretation: This can involve gathering information from multiple sources but, most importantly, interpreting the information to suit it to the skill level or common platforms for your community, and/or adding your own introduction to place the information in context. When writing an advisory, list the important information in the first paragraph: who is vulnerable, whether the problem is exploitable remotely, what the damage is, and the immediacy of the threat. Then discuss how to fix the problem, mentioning any side effects of the fixes. IEEE 1044 describes ways to describe software anomalies in something like these terms.
- Investigation: Be clear about why you are investigating a problem: to better understand the problem?

because you intend to notify the vendor? to check or create patches and workarounds? You can investigate based on incident artifacts (e.g., files left on a compromised machine), source code if available, or test systems (which are not on a public network!).

- Coordination: This refers to working with vendors to resolve a problem. This can be tricky; trust must be built and is easy to lose, and the motivations of the parties may compete. Vendors don't want bad publicity, but our constituency needs patches to prevent incidents, and so do other sites. (But will any publicity of our efforts increase the risks to those other sites?)

The presenter is a partner in the TRANSITS project, a European initiative to provide training on issues related to the provision of CSIRT services, and has documented the mechanics of how to write an advisory. The relevant links are, respectively:

http://www.ist-transits.org/
http://www.ja.net/documents/gn_advisories.pdf

Advisories

Michael Caudill, Cisco Systems Ltd, UK

At Cisco, an advisory usually starts with some kind of notification of "bad news" from an independent researcher or a customer. It is a good idea for the vendor to send some kind of response within 24 hours, because otherwise the person will feel that their message has not been heard/read; Caudill recommended using a PGP-signed reply. Nevertheless, vulnerability reporters should keep in mind that the vendor may be working different office hours, have different national holidays, or be a small shop that has only one person receiving vulnerabilities, so it is reasonable to allow a week or so for the vendor to respond.

Once the vendor has received and acknowledged a vulnerability report, it needs to reproduce and verify the problem: What exactly is the problem? Determine workarounds, and whether they are effective and feasible. Find the fix. Determine whether other vendors are affected. Determine if the problem deserves an advisory, based on ease of exploit, impact, and so on.

The "reproduction and fixing" stage may take from a few hours to a few weeks, depending on how much information the vendor has, the complexity of the setup, and, of course, the difficulty of debugging. Also, teams may be small and working on other cases already.

The advisory is the vendor's official response to notification of the vulnerability. It is best to prepare the advisory before you need it, in case events force you to publish prematurely. Use an informative title, the status (draft, interim, final), the date in GMT, a summary to be read by management and by techies to determine whether the advisory applies to them at all.

Other information that should appear in the advisory: which hardware and software models are and are not affected; specific configurations that are affected if applicable; what causes the problem; the symptoms of the problem (crash, performance slowdown, etc.); the actual consequences (unauthorized access, DoS, information leak, etc.); who discovered the problem (give credit where due); whether the problem is being exploited, including the names of known exploit tools where applicable; links to other advisories; CVE number. Determine what language(s) should be used in the advisory.

Obtaining the actual bug fix from developers can be difficult, because of the commercial pressures to provide features they are under.

If other vendors are affected, either notify them directly (especially if the number is small) or hand off to a CSIRT coordination center of some kind to deal with contacting the other vendors.

When the fix arrives, check it, and do regression testing to make sure that the fix doesn't break anything else (unless the fix is really trivial). Before releasing all this, make sure, if the fix will require a hardware upgrade, that this hardware is available through the normal channels.

Finish off the advisory with information about the migration path and how to obtain fixed hardware or software. Run the early copy past developers; legal specialists, including export control people for crypto; the PR department; and selected groups of technical people – for example, the original reporter of the vulnerability.

Decide when to publish the advisory. Do we have to wait for other vendors? What day of the week is it in other parts of the world? (Of course, if there is active exploitation, release as soon as possible.)

The advisory should use a vendor-independent format (text, HTML, PDF) and should be cryptographically signed. It makes sense to release an advisory internally first so that, for example, tech support knows what the customers are calling about! However, take into account the possibility of leaks, so pre-release only on a need-to-know basis, and/or don't allow much lead time; people need to prepare themselves, but too much time will increase the likelihood of leaks.

Once released, the advisory (at least the version on the Web) needs to be kept up to date; there may be corrections and additional information. When making changes, it is a good idea to update the revision number and keep a revision history so that people can decide whether they need to re-read the document.

KEYNOTE

A Global Culture of Security

Marcus H. Sachs, US Department of Homeland Security, USA

A "culture of security" is developing around the world which involves everyone, IT people and end users alike, and which is parallel to the "safety mind-set" that makes us wear seat belts in cars. The use of computers and networks puts any country at risk; that vulnerability must first be recognized before it can be addressed. In the early 1980s, AT&T ceased to have a telecommunications monopoly. Since then, the US telecommunications network is no longer domestic, terrestrial, and circuit-switched; there is a diversity of circuit- and packet-switched technology, terrestrial, satellite, and wireless, supporting voice, data, and other communications.

In the late 1990s, a military study determined that the Department of Defense could be reached from the Internet, and could be attacked through that route. Somehow, that information was at the time considered to be of only theoretical interest. A few months later, DOD computers became the subject of Net-based attacks. That got the attention of the leadership, and a task force was created to coordinate the defense of digital networks. At the end of that decade, there was a lot of cybercentric effort because of the impending Y2K, though for a while physical sectors of the infrastructure were somewhat neglected.

September 11, 2001, made it clear that the "physical" sector ought not to be ignored. When the two towers of the World Trade Center and some adjacent buildings were destroyed, the telecommunications redundancy that had been provided for the New York Stock Exchange turned out to be inadequate. Although connectivity had been purchased from several different companies, using several physical routes and going to two separate central offices, purchases and mergers among those companies had had the result that most of the "redundant" connectivity went through the same fiber bundles to the same C.O. In most large cities, telephones are connected to a single C.O., with little redundancy. On September 11, phone service in New York basically failed. Interestingly, people were still able to use the Internet to communicate. It was also found that the co-location of the various different utilities (water, gas, steam, electrical power) constituted a vulnerability.

The US government decided that "security" had been divided into too many little offices; the Department of Homeland Security was created to bring these functions under one umbrella. This department published two major documents in February 2003: "The National Strategy to Secure Cyberspace" and "The National Strategy for the Physical Protection of Critical Infrastructures and Key Assets," which are available from http://www.whitehouse.gov/homeland/. The Cyberspace Strategy is intended to be modular and to change as needed.

Why is it important to secure cyberspace? The USA is fully dependent on cyberspace, and the range of threats is huge, from script kiddies to nation-states. Recommendations include addressing vulnerabilities, as opposed to threats, because threats are constant and everywhere. Also, the government alone cannot secure cyberspace – individuals and industry must participate. The speaker went on to list and describe many initiatives being taken by the US government, and he listed requirements for a secure Internet: accountable addressing (such as IPv6); dependable network services (routing, DNS); trustworthy software; authenticated user services (Web, email); a working public key infrastructure; networks built to be secure from the start, not as an afterthought; the adoption of "best practices"; protection of and from "clueless users" making mistakes; the certification of network engineers; a mechanism for information sharing about computer security; and agreements between nations with respect to cybercrime.

TECHNICAL SESSIONS

Worst Fears/Workings of a Worm

Roelof Temmingh, Sense Post, South Africa

There are over 1 million networked computers on Earth. The Internet covers not just the Western world, though the vast majority of connected computers are in the Western world, and Internet-based services include not only email and the Web, though those are the major services. Most of the computers are running Microsoft products. About 90% of Web servers run either Apache or IIS, and 96% of browsers are MSIE. The state of security ofn those products is not reassuring. Many vulnerabilities have been revealed in the past few years and have been exploited by some of the major worms. In fact, while the recent worms have exploited previously known vulnerabilities, most had no direct malicious payload or had an effect limited to DoS, and in most cases only a small percentage of targeted hosts were infected; nevertheless, their effect was large.

What, then, is the worst-case scenario? Imagine a group of 15 or so programmers working for six months in isolation.

Phase 0: Perform reconnaissance work (scanning) to find vulnerable Web servers (assume that we have prepared an exploit for an as-yet-unpublished vulnerability that will break Apache and IIS). Pick the 1000 busiest servers.

Phase 1: First we stealthily infect a thousand machines that are important, high-traffic Web servers (e.g., CNN). These "master servers" have huge log files and are constantly under attack anyway, so

our attack has a better chance of remaining undetected, which is very important at this point. The code we inject into our victim "master servers" does DNS lookups of random names using DNS servers on predetermined controller hosts, and the return values of these DNS lookups contain commands for the master servers to execute. We send each master 1/1000 of the list of IP addresses of all vulnerable Web servers as determined during phase 0.

Phase 2: We now infect as many machines as we can, still stealthily. We send a signal to each master to inject attack code into random Web pages (but not the home page) of the Web server it has infected. Therefore, when a browser downloads that page, it also downloads the client version of the attack code. Using these "client servers," we do reconnaissance on the networks they are on (possibly internal networks not directly connected to the Internet), remove any antiviral software, collect any passwords we can find, and take control of any infrastructural machines (such as routers) that we can.

Phase 3: In a "pre-meltdown" phase, at a predetermined time (because communication with infected clients may be impossible), all "clients" look for vulnerable Web servers on the inside of that network and infect them so that they too will now carry the "browser worm," at least for the next three hours. At the same time, all "masters" carry out the infection of their predetermined lists of externally connected vulnerable Web servers.

Three hours after the above (which is short enough that it is unlikely that human intervention can take place in time to stop the process), the "clients" start destroying the local machine and network; they send DoS packets to any hosts that could not be infected, insert random bytes into data and text files of all kinds, corrupt or destroy the BIOS, and pop up a box asking users to call their local help desks, thus swamping the help resources and perhaps even the phone lines. At the same time, the "masters" remove all Web content, and start DDoSing Microsoft and Apache (to inhibit their supplying any patches), the root DNS servers, and random other sites.

Long-term data destruction (e.g., of data on backups) was also discussed, but it is a bit more involved.

The results of such a worm would effectively be total chaos. To prevent such a worm from succeeding, several measures should be taken:

- DMZ: Isolate outside-facing Web servers; don't allow machines in the DMZ to make connections to the inside.
- Tight filtering: A Web server should never initiate connections. If it is unable to initiate Web connections, it cannot spread a worm that way, even if it is itself infected.
- Internal segmentation: In case part of the organization gets infected anyway, the other parts are protected if the internal network is segmented by packet filters.
- Filter any third parties that have connectivity to your network.
- Use personal firewalls on user PCs as a last line of defense.

Automatic Exchange of Incident-Related Data and Its Application in CSIRT Operations

Klaus-Peter Kossakowski, Presecure Consulting GmbH, Germany

eCSIRT.net has a project whose goals are to improve the exchange of incident-related data and to foster improved cooperation among CSIRTs. For this, all groups involved have to agree on the meanings of words so that statistics are meaningful and a shared knowledge base is possible. The project also includes the provision of actual services related to this information, including "out-of-Internet" alerting so that information can get through even when the Internet is non-functional.

A code of conduct binds the participants in the information exchange, and covers issues of cooperation, protection of intellectual property rights and confidential data, and contributions to the goals of the project.

The common language is based on IETF initiatives Intrusion Detection Message Exchange Format, or IDMEF (to send attack information directly from the sensors), and Incident Object Description Exchange Format, or IODEF (for local CSIRTs to send a more high-level view of an incident). A Web form will allow constituents who don't yet support IODEF to send IODEF objects. IODEF will also be used to send information to local CSIRTs. IDMEF incident data can be included within IODEF incident descriptions.

eCSIRT.net has a clearinghouse function, to share as much information as legally can be shared; this is a low-priority task which occurs after incidents are closed, at which point they can be processed for statistical purposes. The output will be tailored to the recipient, where participating CSIRTs will get more detailed information, and the general public will get just an overview.

Three types of statistical information will be collected:

- Type 1 data are related to the workload of each CSIRT: number of attacks reported; false positives; systems attacked and affected; time spent analyzing, responding, documenting; and so on.
- Type 2 data concern incidents themselves. This information will be aggregated, and any identifying

information removed before it is presented to anyone.
- Type 3 data pertain to Internet events (which are not necessarily intrusions). These events will be monitored using automated techniques such as Argus (traffic monitor), honeypots, and so on. The event information will be accessible to constituents via HTTPS and user certificates.

The incident (type 2) information is highly controversial, because of trust and confidentiality issues; these issues are being addressed. There is potential that type 3 (event) information could have online processing resulting in alerts that can be sent to constituents.

There is technology to send out encrypted mail, to make phone calls and faxes, and so on. The idea is to free humans to analyze problems instead of tying them up in the mechanics of the transferring and storing of information.

Request Tracker for Incident Response

John Green, JANET-CERT, UK; Jesse Vincent, Best Practical Solutions, USA

RTIR is a tool for incident handling, which claims to be usable, cross-platform, open source, extensible, securable, and supported.

RT (the base software for RTIR) was designed to track issues of any kind. It gets used for bug tracking, help desk and customer service, network operation, "to do" lists, and so on. In its bare form, it does get used for incident response, but it is not ideal for this use. It is Web-based, and the client side is designed to work with just about any browser.

RTIR is an extension to RT, which uses the designed extension mechanisms; so, for example, it is possible to upgrade RT without having to "repatch" the extension. It is possible to add functionality, change the user interface, and so on. It should be run using HTTPS to improve security.

RTIR adds these functions to RT:

- An "incident" object ties together the various reports that might come in about a single incident and various actions such as blocking networks and performing investigations.
- Incident response team–specific workflows, for example, automatically opening incidents to notify the people responsible for each of a list of IP addresses about a vulnerability discovered while scanning.
- "Clicky" metadata extraction and tracking (to get more information about IP addresses through such utilities as whois, traceroute, etc.).
- Integration of whois information.
- Separate email threads for separate conversations about different tasks or components of the event.
- High-level overviews.
- Better searching tailored to incidents so that multiple events can be correlated.
- Simple scriptable actions.
- New reporting functions.

More information can be obtained from *http://www.bestpractical.com/*.

Communication in Software Vulnerability Process

Tiina Havana, Juha Roning, University of Oulu, Finland

Reporting software vulnerabilities is central to software development, but the communication process is problematic. In 2002, a study was done where vulnerability reporters and the receivers of such reports were questioned. Recipients reported contacting reporters more than the reporters believed that they were contacted.

The values and beliefs of the two parties differ. While they agree on the importance of security, precision and accuracy, and non-malfeasance (avoiding harm to others), reporters value public benefit and the public's right to know more than do the receivers, while the receivers place a higher premium than do the reporters on the avoidance of "FUD" (fear, uncertainty, and doubt).

Only just over half of the receivers passed on the bug information to their developers to prevent further occurrences of that type of bug.

One-third of the receiving organizations have a proactive publicity strategy for cases where there is a publicity crisis concerning vulnerabilities in their products, and one-third of them have PR personnel who are familiar with vulnerability issues and who have direct media contacts.

The vulnerability-reporting communication seems too often to be one-way; two-way symmetrical communication is needed. A dialog between the parties would improve mutual understanding, and vulnerability reporting policies would also be helpful.

PANEL DISCUSSION

Ask the Experts

Cory Cohen, CERT/CC, Carnegie Mellon University, USA; Robert Hensing, Microsoft, USA; Michael Warfield, ISS, USA; moderator: Roger Safian, Northwestern University, USA

Q: Concerning Microsoft's recent acquisition of an anti-virus product vendor, is Microsoft planning to automate anti-virus download-and-security-patch management into the same agent?

Rob: Microsoft is working on an anti-virus product of its own. It is also working with VIA (the Virus Information Alliance), and working on other security-related services geared toward home users. I cannot comment on the details.

Q: In the case of an organization with no real security other than that provided by volunteers, what "glory words" can be presented to financial folks to persuade

them to devote resources to a security team?

Rob: To justify an IRT, use threat modeling: What am I trying to protect? What is it worth? A model called STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Damage Potential, and Exploitability) assigns dollar values to each of the named items. Chapter 4 in Writing Secure Code walks through the STRIDE model.

Andrew Cormack: Universities don't work on dollars and cents – list assets, liability.

Q: For home users and the general public, a major problem is spam. Break-ins on DSL-connected hosts are often motivated by the desire to install spam distribution agents. Various government organizations are interested in stopping this because it costs a lot of money. As security people, should we do something radical to the email infrastructure to address this issue?

Mike: Honeynet analysis of scanning for open proxies (squid, socks) shows that these scans are done mostly by spammers wanting to inject their spam. These guys have crossed the line at that point into illegal activities, therefore prosecutions need to be done. Spammers are trying to sell something, so they must leave ways to track them back. Our responsibility as security people consists of hardening email systems, implementing authentication between servers, and locking down open relays.

Cory: The growth of really usable cryptography could help. Membership in mailing lists should require use of cryptography to participate in them.

Roger Safian: The issues of someone abusing your resources versus someone just sending spam should be kept separate.

Rob: Technical approaches to this problem are seen here at conferences. Microsoft is chasing spammers legally' We need high-profile prosecutions to discourage this type of activity. Microsoft is suing 13 spammers, of which 11 are in the USA and 2 are in the UK. 60% of mail coming to Hotmail is spam!

Q: Lots of spam is fraud (eBay fraud, trying to get credit card information).

Mike: eBay and Pay-Pal frauds get much press coverage, but those make up less than 1/10 of a percent of the incoming spam, while one-third of it is for Viagra and other "personal enhancement" (!) technologies. Prosecutions need to proceed on the fraud front; this type of activity must become unprofitable.

Q: Regarding security advisories, is there any chance that advisory formats will converge so that automated means can be used to disseminate them, or their existence, to our constituents?

Mike: Advisory formats are evolving, but no wholesale conversion to a single common format should be expected. We may see two or three different versions of the same information (text, Web, XML), of which XML is most amenable to automated processing. But advisories must go through marketing and PR departments, which are less amenable to technical standards!

Cory: We are open to exploring standardized formats, but this is less important for advisories. We are more interested in a standard format for the exchange of vulnerability information, which can then be customized with text for advisories.

Rob: Microsoft released two new bulletins just today. After the advisories have gone through the legal people, they are mangled! If a standards body were to come up with a standard format for advisories, I would encourage Microsoft to probably play along. But Microsoft bulletins are already huge; a list of minimal information would be helpful.

Q: We have seen suspicious packets with a fixed window size. These were discounted by CERT/CC, while ISS went further and gave meaningful data. Give us some idea of your internal processes.

Mike: ISS got caught with its pants down on this one. We had noticed peculiar traffic, deferred looking into it for a week, then looked into it after incident reports had appeared in various other sources. At that point, we started detailed investigations, which are still in progress [as of 06/25/03], but postings have been made.

Rob: We [Microsoft] saw the traffic. No one could figure out its origin, and this caused great concern: Was it the first big kernel-mode Windows worm? [It turns out that this is probably not the case.]

Mike: This is a classic example of the fact that our business gets interesting with the words "I never saw it do that before!"

Q: What are the pros and cons of protocol- and signature-based IDSes? What are some of your favorite IDSes?

Cory: As a member of CERT/CC, I have no favorites, but we did some research on IDSes to assess their ability to really describe vulnerabilities using signature languages. We were disappointed about what could be done to really describe vulnerabilities. A lot of signature-based systems could not describe a recent Kerberos vulnerability, for example. Therefore, I have doubts about signature-based systems.

Mike: Marketing people want us to believe that the two kinds are very different. Protocol-based engines need more horsepower. Therefore it makes sense to try for the cheaper signature-based methods first to skim off 80% of the problem; then do protocol analysis to catch more; then, finally, there's the holy grail of "anomaly detection," which does not really exist yet. Each technique

has its own domain of applicability; all of them should be used.

Rob: I'm more interested in host-based than network-based IDSes; the OS needs to get more intelligent. Tripwire, for example.

Q: Are there signature development classes, tips on developing signatures?

Mike: Get in good with the Snort guys, who are doing it all the time on our mailing list.

Cory: I found it difficult to match up individual signatures from various systems.

Roger Safian: We just completed testing eight IDSes of multiple kinds. All had plusses and minuses. Sales people are overzealous in their claims.

Q: What about intrusion prevention technology?

Mike: I have similar comments as for "protocol versus signature": there is a big area of overlap between technologies, and fuzzy definitions of them. Many vendors talk about putting things in midstream and blocking based on findings. There are concerns about false positives; we need a bit more track record on these devices.

Rob: Intrusion prevention is another way to say "host hardening": You can't attack what isn't there! I've been told that Windows listens on too many ports, that we cannot turn the services off. But Win2K has IPSec, which can be used to block access to ports.

Roger Safian: SANS cornered Gartner. It turns out that intrusion detection devices that are actually deployed are generally not used to block traffic!

KEYNOTE

The European Initiatives in Network and Information Security

Andrea Servida, Information Society Directorate of the European Commission, Belgium

Why Europe needs to act on security: 75% of European companies had no security strategy in 2002. It seems that underestimating core business risks is responsible for the low level of IT security investment in most European companies; there is a lack of awareness of the issues. The paradigm for security is changing from "security through obscurity" to "security in openness," but this openness and sharing of resources is a challenge to manage.

Toward a European-integrated approach to security: International cooperation is needed and is occurring in the following areas:

- Economic, business, and social aspects of security in an information society: These include business opportunities and growth, individual issues such as privacy and the protection of minors, dealing with the digital divide, and long-term preservation of knowledge and culture.
- Cybercrime, "homeland" security.
- External security and defense.
- Security research.

The role of the Commission: The European Commission proposes and orchestrates the development of a regulatory framework, for example governing electronic signatures, data protection in electronic communications (consent to collection of data, use of cookies), information, and network security. The Commission also launches policy initiatives, in particular to promote the development of various technologies or approaches.

eEurope 2005 and ENISA (European Network and Information Security Agency): eEurope 2005 build on the progress made by eEurope 2002: Internet penetration in houses has doubled since then, there is a telecommunications framework in place, and there is a very fast research backbone network. There are projects which affect CSIRTs, such as the European Information Security Program, which aims at providing SMEs with the IT security solutions needed to develop their e-commerce business. eE2005 aims to:

- Establish a cybersecurity task force by mid-2003. ENISA is not itself a CSIRT, but would have a facilitating role, coordinating existing capabilities and resources in member states.
- Develop a "culture of security" by the end of 2005 (develop good practice and standards).
- Secure communications between public services.

Ambient intelligence (wearable computers, computers inside our bodies) raises new security issues. Today's technologies are already pervasive and intrusive, and have huge interdependencies which are not being managed well. The challenge for tomorrow is to develop a respectful approach; the ethics of privacy will be a key element in an information society.

TECHNICAL SESSIONS

Introduction of the APCERT: New Forum for CSIRTs in Asia Pacific

Yurie Ito, JPCERT/CC, Japan

There is now a new forum for cooperation among CSIRTs in the Asia Pacific region: APCERT.

Most CSIRTs in the AP region do direct incident handling and coordination as well as issue warnings and alerts, technical bulletins, and so on. However, there is little participation from IRTs in industry. Many CSIRTs in the AP region do communicate directly with each other, to share observations, data, and technical information and to contact sites involved in an incident. While there are

commonalities between the various CSIRTs, such as a narrow block of time zones and IP address blocks, there are differences in the technological maturity of the various participants.

There was a desire to coordinate the interaction between the AP CSIRTs. A working group was formed in 1997, whose core members were three large teams: CERTCC-KR, SingCERT, and JPCERT/CC. Over the years, it was decided to create APCERT, which was established in February 2003. Its objectives:

- Share security information among members.
- Handle security issues on a regional basis.
- Support the establishment of CSIRTs in countries not yet covered by such services.
- Collaborate with other regional frameworks, such as FIRST and TF-CERT.

There are currently 15 full members. APCERT's activities include an annual conference (APSIRC), and working groups on accreditation rules for APCERT membership, on training and communications for CSIRTs, and on financing the APCERT effort.

The APCERT involves other players, such as users, system integrators and operators, regulatory bodies, the insurance industry, law enforcement, and the technology development and engineering communities. It encourages these players to cooperate and communicate with each other; it facilitates mutual trust and information sharing.

Privacy Incidents on the Rise: Taxonomy and Response

Lance Hayden, Cisco Systems Inc., USA

What is privacy? As incident responders, we are called upon to take a "first responder" approach to privacy breaches, whether or not the direct causal link to computer security incidents is evident.

What is under attack? There is an evolution away from "computer security," where we are protecting information but we don't necessarily need to know what information it is we are protecting. We are now realizing that this information has a meaning, and can be used to cause serious damage to people and institutions. Attackers break into systems in general because they are looking for information; privacy is an extension of what CSIRTs have been doing as part of their jobs.

Identity theft is turning into one of the most prevalent crimes of this century. Criminals need anonymity (in the form of an identity other than their own), and stealing an existing identity that has a history is much more useful to them than creating a fake identity, which can be found to have "popped into existence" recently and can therefore be flagged as fake. This has implications beyond someone losing money from a stolen credit card, and we can expect civil and criminal liabilities for "enablers."

As the concept of "computer security" has evolved to "information security" and then to "privacy," the people responsible for working on securing that information have grown to include not only sysadmins, but also CIOs and, finally, individuals, who must safeguard their own information. There is a plethora of laws affecting privacy at the national and local levels.

The author suggested four basic categories (a taxonomy) of privacy breaches:

- Malicious attack: often preceded by security breach
- Process breakdown: security processes fail to protect
- Human or system error: not malicious, but still damaging
- Other

The following item apparently appeared on Slashdot the day of the presentation: The Palo Alto Unified School District had a wireless network which was not secured properly. A reporter for the Palo Alto Weekly parked in the parking lot and was able to pick up, for each student, full names and addresses, pictures, and in one case a psychological evaluation. While this could be characterized as a malicious attack, in fact errors in system configuration made the task of obtaining this information trivial.

The speaker described several more privacy breach cases, including one in which used computers were sold with their disks incompletely wiped, so that personal information about hospital patients (for example) was compromised.

Recommendations: Be prepared for the implications of privacy breaches. Apply not only deductive reasoning (what happened to cause this breach?), but also inductive reasoning (anticipating the impact of information loss).

Honeynets Applied to the CSIRT Scenario

Cristine Hoepers, Klaus Steding-Jessen, and Antonio Montes, NIC BR Security Office, Brazil

The Brazilian CSIRT set up a honeynet with the objectives of monitoring current attacks and intrusions, collecting data about opportunistic ("script-kiddie"-type) activity, developing new tools, and evaluating the usefulness of honeynets to CSIRTs. Requirements for the honeynet included low cost and reliability, as well as a high-quality data control mechanism. The team also wanted to make sure that it could prevent the use of the honeynet as a launching platform to attack other sites. Therefore, free software was used, and data was stored in "libpcap" format to facilitate its analysis with existing tools.

This team's honeynet started operations in late March 2002.

The honeynet's topology includes an administrative network, whose functions are to prevent outgoing attacks, to log activity, and to store artifacts and disk images of the honeypots. The honeypot segment includes several honeypots running different OSes.

Technologies used to control outgoing data (to prevent attacks from within the honeynet) include firewall rules (e.g., to block spoofed packets), outgoing traffic normalization (to discard invalid packets), a tool called "sessionlimit" (which can limit outgoing traffic based on fairly sophisticated state-based rules), bandwidth limitation to prevent DoS attacks from the honeynet, and an outgoing filter ("hogwash") to block traffic based on contents. Alerts and summaries are produced to report on activity in the honeynet.

Activities seen in the past year included lots of IRC traffic, a lot of worm activity (some new worms were captured), and denial-of-service attempts. Several new rootkits and exploit tools were captured. Statistics were produced on the top scanned ports (FTP, SSH, Telnet, and portmap were at the top). Also, many scans for open relays and open proxies were seen on ports 25, 1080, 3128, and 8080.

Worm activity concentrated on ports 80, 443, and 1433. There is still a lot of Nimda and Code Red activity, which means that many compromised hosts are still active!

The origin of the problematic traffic was graphed by country; for most cases, the US was at the top, though not in as large proportion as the US's presence on the Net. Back-door access and language of IRC conversations point to Romania as a major source of malicious activity.

By maintaining a honeynet, a CSIRT can provide an additional source of data to help understand what's going on in a particular set of incidents, and can pinpoint and notify compromised machines of constituents if they show up in the honeynet logs. This work is also a great source of training material for log and artifact analysis and forensic methods, and can be used to capture attack tools that would otherwise be only in the victim host's memory, since it is possible to capture full network traffic on the honeynet.

An Internet Attack Simulator Using the Extensions of SSFNet

Eul-Gyu Im, Jung-Taek Seo, and Cheol-Won Lee, National Security Research Institute, Republic of Korea

Because of the increase in Internet attacks, there is great demand for research on them and their effects. However, it is not always possible to study the attacks on a production network, for obvious reasons. This is why network simulators are useful.

SSFNet (Scalable Simulation Framework, network module) is a freely available network simulation tool which has a process-based discrete event-oriented kernel. The authors added extensions to SSFNet: a firewall and a packet manipulator. They then performed experiments with this setup; for example, they simulated a "smurf" attack in a network of 13,000 clients, 40 servers, and 270 routers, and were able to show the degradation of ping response times as a function of the number of subnets participating in the attack; it turns out that response times degrade drastically when 12 or more subnets are involved in the attack.

Policy-Based Configuration of Distributed IDS

Olaf Gellert, Presecure Consulting GmbH, Germany

IDS is a security component which monitors system events for unwanted behavior, using the following methods:

- Anomaly detection: As a first step, the IDS gathers statistics about normal behavior, and as a second step, it generates alerts on unusual behavior.
- Misuse detection: The IDS compares events against patterns of known attacks.

There are different kinds of sensors:

- Network sensors (NIDS) are components which inspect network traffic. One sensor per subnet is needed, and there is no feedback from hosts.
- Host sensors (HIDS) require one or more sensors (one per host), but they permit direct access to information.

An IDS also needs analyzing components to collect the data from the sensors, to take action on logged data, and to compile statistics. Management consoles are the front-end for visualization of logged data.

All IDSes suffer from false positives, false negatives, and often offer no explanation of an anomaly. With distributed IDSes, a large amount of generated data adds up: It's important to correlate all alerts for one attack into one single alert. With lots of sensors, the problem of configuring all those sensors becomes significant.

There is a diversity of different specification formalisms used for the configuration of IDSes, and different types describe different events. There are also configuration differences, even among several sensors of the same type, depending on their placement. In addition, anomaly-based sensors require fre-

quent updates to their attack-recognition signatures.

There are several possible solutions to this configuration complexity, including some unacceptable solutions such as using only one vendor and/or only one type of sensor. There have been attempts to specify a single configuration language for all types of sensors, but this does not solve the problem of different configurations being required based on each sensor's placement within the network. The speaker suggested a solution: specify only a policy and generate rules automatically based on this policy.

The policy description suggested contains users, hosts, services, access by source/destination, and restrictions based on these combinations. In addition, one needs a database of rule sets to describe known attacks (for signature-based IDSes), a database of assets (existing systems, installed software), and a network topology database of some kind (not implemented yet).

A policy is used to generate a rule outline (a description of allowed services), based on which we use the asset database to generate rule specifications for each IDS based on their placement. Finally, we use the contents of the signature database (where applicable) to generate particular rules for each sensor. We can also set the severity for sets of events: for example, distinguishing between "alert for this event" and "just collect stats on this event."

The results of this work suggest that it is possible to configure all sensors centrally. Specific rules for each sensor reduce the number of false positives, and we see improved accuracy on the severity of alerts; it becomes easier to update the rules. A side benefit is that the assets database can be used for other purposes.

The maintenance of the asset database takes time, though this maintenance could be supported by automatic processes, using the output of the IDS, for example, or using scanning tools. Actual updates might still require manual confirmation, as might the signature database, though in that case the IDS vendors can help. Some unresolved problems: how to update rules without restarting sensors? how to introduce advanced topology information?

The status of this work: A tool to configure IPtables and Cisco access lists now exists (written in awk), and nearly completed work in object-oriented Perl is being done for handling subset-of-policy objects.

The Stealth File Integrity Checker

Frank Brokken, University of Groningen, the Netherlands

STEALTH (SSH-based Trust Enforcement Acquired through a Locally Trusted Host) is a file integrity scanner that has the advantage of being stealthy.

STEALTH's mission is to ensure the security of our computers (integrity, availability, confidentiality of information stored). Intruders may modify the integrity of the information on a host; our lines of defense include denying access when an intrusion attempt is detected, keeping software patches up-to-date, monitoring system logs, and, finally, knowing when relevant information is altered. This last is the goal of a file integrity checker.

File integrity checkers work by creating a "fingerprint" of the current state of a host, then detecting modifications of that fingerprint. The problem with the traditional approach of storing the fingerprint on the monitored computer itself is that the fingerprint itself is therefore vulnerable to intruders. The usual solution, to keep this state on read-only media, makes updating the fingerprint difficult or costly. The solution is to store the fingerprint on another computer, out of reach of the intruder, but in easy reach of the sysadmin.

The stealth "master" itself is not connected to the outside Internet, but can make SSH connections to its monitored clients. The fingerprint is stored by the monitor; there are no logs on the clients.

STEALTH itself uses standard software like "find" and "md5sum," so it is highly flexible and adaptable. The timing of STEALTH runs is unpredictable. Because the actual file signature calculations occur on the client, the monitor can be any old hardware; resource requirements are small. STEALTH can be obtained at *ftp://ftp.rug.nl/contrib/frank/software/linux/stealth/*.

KEYNOTE

Cyber Security in Canada

James Harlick, OCIPEP, Canada

Canada created OCIPEP (Office of Critical Infrastructure Protection and Emergency Preparedness) in February 2001 to enhance the safety and security of Canadians in their physical and cyber environments. The mandate has two components:

- Protection of critical infrastructure: physical and cyber components of the energy, utilities, communications, services, safety, and government sectors
- Emergency preparedness for all kinds of emergencies

Currently, the critical information structure is highly interdependent and has numerous vulnerabilities (e.g., a worm knocked out 911 service in part of the US because they were using VoIP). Canada has pledged itself to be the most connected nation on earth by 2006, which puts its infrastructure at great risk.

Canada's cyber-security framework contains four ways to reduce risks:

- Strengthen policy framework: Government departments and agencies are required to report cyber-threats and incidents to OCIPEP, and a framework is created to support information sharing and protection among various jurisdictions.
- Enhance readiness and response: protect (issue alerts and advisories), detect (coordinate identification and analysis), respond (establish incident response centers), and recover (provide incident impact analysis, technical assistance).
- Build capacities: This includes training and education (CSIRT training, training on malicious code analysis, IDS data analysis, assessment of vulnerabilities) and R&D (coordination with funding councils of the government, direct research projects).
- Build partnerships: It is necessary for partnerships to be formed internally, among the federal and provincial governments as well as critical infrastructure owners and operators, and externally, with other governments and CSIRTs. Already there is a daily "health check" information exchange between OCIPEP and the provinces, and there is a weekly conference call with critical infrastructure owners and operators in the private sector.

TECHNICAL SESSIONS

Multi-level Monitoring and Detection Systems (MMDS)

Madhavi Latha Kaniganti, D. Dasgupta, J. Gomez, F. Gonzalez et al., University of Memphis, USA

The presenter described an intrusion detection system (MMDS) which is an agent-based approach to monitoring and detecting attacks. The design is a hierarchy of specialized agents, with a fuzzy decision support system used to generate rules for attack detection. There are four types of agent:

- Manager agent: coordinates work and information flow.
- Monitor agent: gets information from sensors (a monitor agent should run on each host being monitored).
- Decision agent: uses information to decide what to do.
- Action agent: generates alerts and heartbeats. There is a GUI that can show graphs of various measured parameters.

The decision agent, which is the heart of the system, is based on fuzzy logic (fuzzy sets with "degrees of set membership" between 0 and 1). Instead of hand-coded fuzzy rules, rules are generated automatically by "training" the decision agent with genetic algorithms, under normal conditions and attack conditions.

Parameters monitored by MMDS include network activity (sent and received bytes and packets), user activity (logins, failed logins, number of users logged in), process information (total number of processes, number of root-owned processes, and number processes in various states), system parameters (physical and virtual memory in use), and MAC-level network data (sequence numbers).

Once the decision agent had been trained to recognize three attacks (SSH hack, nmap scan, and MAC spoofing on a wireless network), the researchers claimed very good results for detecting those attacks under test conditions.

Fire Your Firewall

Jan Meijer, Hans Trompert, SURFnet/CERT-NL, the Netherlands

The speaker's intention was to show that firewalls cause more problems than they solve (which is not to say that anyone has the right solution). The Internet (in 2002) was composed of 11,000 networks with 34,000 peerings, yet despite this complexity, it works! The use of firewalls tends to interfere with the proper functioning of the Net in many ways.

Firewalls break the Internet by misconfiguration. For example, Path MTU discovery, which permits the fragmentation of packets where necessary so they can reach a network with a "smaller" MTU, depends on particular ICMP control messages getting through – yet people do filter them (by filtering "all ICMP"). Other examples where a misconfigured firewall breaks things include jumbo frames, fragmentation, and certain applications, such as FTP, IRC, H323, IPSec, IPv6, and multicast.

Another reason firewalls break things is the added complexity they introduce: Because communication is no longer "end-to-end," it becomes difficult to find errors. Things no longer "just work" a lot of the time. The increased number of machines, people, and procedures involved means more opportunity for things to go wrong.

The time spent working around firewalls is time wasted (and sometimes workarounds are not available). For example, H323 videoconferencing requires four extra machines to work through a firewall!

Firewalls are single points of failure and will, of course, contain errors (since they are developed using the same processes used for other software!), and yet we place our trust in them.

Firewalls limit network speed by creating a bottleneck: Will firewalls, especially those which must traverse the protocol stack, keep up as networks speed up?

Firewalls consume scarce resources by requiring staff, equipment, time, and money to acquire and maintain. On a related matter, firewalls create bureaucracy: policy, carefully kept rules, lots of administration.

Firewalls create a false sense of security: There is usually traffic which works around or tunnels through the firewall (for example VPNs), and lots of traffic on legitimate protocols (email, Web) is still dangerous. Firewalls provide no denial-of-service protection. Too many configurations filter only inbound traffic; in this case, a "call-back tunnel" will get around the "problem," and local users do set these up. Even if there is a firewall, it is still necessary to patch, use end-to-end encrypted communications, switch off unneeded services, etc., measures which are too often neglected when there's a firewall.

Incident Response and the Role of Law Enforcement

Kimberly Kiefer, Computer Crime and Intellectual Property Section of the US Department of Justice

The CCIPS, a 30-attorney section within the Criminal Division of the US Department of Justice, prosecutes computer crime and criminal intellectual property (IP) cases, trains and counsels agents and prosecutors, develops policy on computer crime and IP, provides input on legislation, and represents the USA on international bodies which address computer crime and IP issues.

Security incidents are generally underdetected and underreported. Some reasons for not reporting incidents include the fear of negative publicity and competitive disadvantage, uncertainty as to whether law enforcement is interested or able to deal with the crime, not wanting to "challenge" crackers, and not knowing who to call or what is worth reporting. The speaker reassured us that law enforcement tries to be discrete with information and does not seize victims' computers; the victim does not lose control but, rather, is consulted closely. Also, the number of law enforcement agencies able to deal with computer crime issues has increased at all levels in the US.

There have been some success stories — for example, the successful prosecution of Mafiaboy (DDoS) and the Melissa virus author.

We are encouraged to report incidents even if the damages do not meet the police's criteria for investigation, because our incident may be linked to others, which collectively do meet the requirement.

Tips on cooperating with law enforcement:

- Keep detailed notes and logs, including records that will quantify the damages caused by the incident.
- Set up contacts with law enforcement before an incident occurs.