A THEORETICAL SUPERWORM

-

APRIL 2002

INSTITUTE FOR SECURITY TECHNOLOGY STUDIES
AT DARTMOUTH COLLEGE

INVESTIGATIVE RESEARCH FOR INFRASTRUCTURE ASSURANCE GROUP

IRIA/ISTS
45 Lyme Road
Hanover, NH 03755
603-646-0700

Executive Summary

This report will explain the current model of vulnerability detection, assessment, and response. The current cycle is as follows:

- Identification of vulnerability
  - Inform security community and/or appropriate vendor and/or press
  - Scan for possible existing exploits of vulnerability
- Development of response, i.e. patches
  - Inform the community of security measures to be taken
  - *Hope everyone takes responsibility for applying patches*
- Observe effects of vulnerability/exploit on unprepared systems
- Learn from observations, develop new strategies and tactics, and identify new vulnerabilities

The creation of a **SuperWorm**, malicious code written to incorporate the most successful features of known worms and other malware, could threaten the economy or national security. The SuperWorm has three tasks:

- The worm's programmer identifies multiple exploitation vulnerabilities that affect a large number of operating systems
- Propagation
- Delivery of payload and damage

The SuperWorm programmer could learn from and utilize the speed of Code Red and CodeRed2, the multiple means of proliferation utilized by Nimda, the ability to incorporate other virus code to be transported with the worm seen with variations of the Klez worm, the ability to access and send files from a hard drive as seen with SirCam, and the destructive payload of Magistr.

The **Incident at XYZ University** provides an example of a network that was established with information sharing and open education in mind. The security features of the network do not provide adequate protection against an advanced threat. Additionally, the limited resources of the systems administrators do not facilitate either the rapid response to vulnerabilities or the collection and analysis of the evidence of an attack. These factors limit the ability of the administrators of this network, and networks similar to this University, to defend their system against a sophisticated, rapid cyberattack.

The early warning system described in the **Early Detection of Active Internet Worms by Metering ICMP Destination Unreachable Messages** section provides an example of a security technology capable of detecting worm activity. Early detection will allow security experts to learn about the technical specifications of the worm, and the vulnerabilities exploited by the worm, and give them time to develop a response to the threat. Additionally, the warning system provides data for further analysis of worm and attacker behavior that will enhance the ability to defend against future attacks.

The **Modified Reverse Proxy Server** (JEANNE) is an example of a security technology that mitigates the threat of worm proliferation on a web server. The technology is a proactive means to address possible threats before they manifest themselves in the wild.

Table of Contents

Introduction

This monthly briefing examines a scenario of a SuperWorm release into the current Internet environment. A theoretical worm description will be provided, and an examination of an actual incident of computer intrusion will be examined to indicate conditions that could facilitate the spread of the worm. Finally, two security technologies that could potentially mitigate the spread or damage of the worm will be described.

The aim of this report is to identify the current model of vulnerability detection, assessment, and response. Ultimately, the goal is to help network administrators in the development of adaptable and comprehensive responses to address vulnerabilities.

There is a tenuous balance between reactive and proactive measures taken to address vulnerabilities to national critical infrastructure systems of the U.S. and other networked computer assets. The current means to address vulnerabilities in software and systems revolves around the cycle of identification and disclosure of a vulnerability followed by the simultaneous development of patches and exploits. The race to infect (exploit) or inoculate (patch) vulnerable hosts eventually settles into an equilibrium, at which point both sides, both attackers and protectors, take stock of lessons learned. The cycle begins again, and often the original exploit is re-introduced with counter-countermeasures to defeat the original steps taken to mitigate the threat. Figure 1 illustrates the cycle of vulnerability detection, assessment, and response.

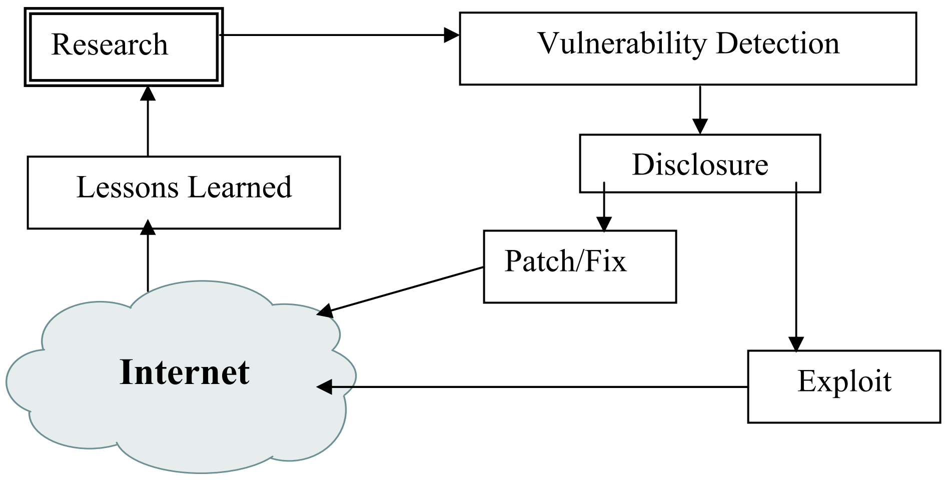

FIGURE 1

Vulnerabilities are discovered daily, and are increasing by the year. Please see **Figure 2** for *Vulnerability Statistics from 1995 to 2001* from the CERT® Coordination Center (CERT/CC), the computer emergency response team of Carnegie Mellon University. Current rates of vulnerability discoveries place a strain on the security resources in place to formulate timely responses. Security is often a secondary concern for both developers of software and technology, and by the companies and individuals utilizing these resources. The response to vulnerabilities tends to be reactive to specific malware (malicious code intended to harm or to destroy data or general system operability) and exploits, relying on early threat identification and proper response formulation to defeat a possible exploit. Although companies and individuals are increasingly aware of the necessity of taking proactive measures to protect their infrastructures, implementation is still dependent upon human diligence and skill in deploying the security measures.

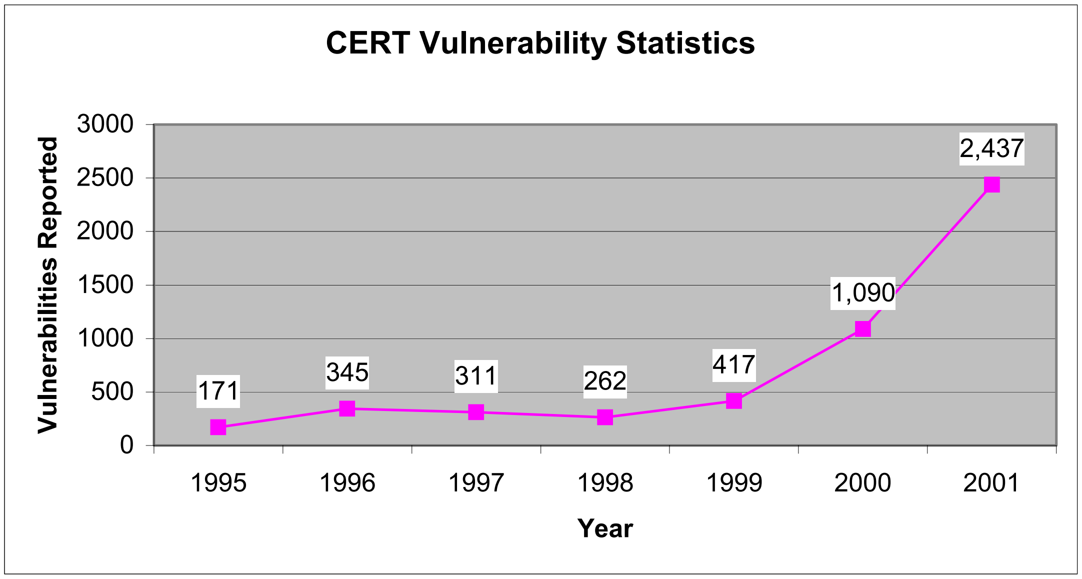

FIGURE 2 (http://www.cert.org/stats/cert_stats.html)

A survey conducted jointly by the Federal Bureau of Investigation and the Computer Security Institute (CSI) in Spring 2002 indicated "ninety percent of respondents (primarily large corporations and government agencies) detected computer security breaches in the last 12 months. Eighty percent acknowledged financial losses due to computer breaches."[1] The rates at which viruses, worms, Trojan horses, and other malware are being developed, the ease of accessing prefabricated exploitation tools published over the Internet, and the common occurrence of detailed vulnerability publications and the means to exploit it prior to the development of a patch or fix, indicates not only a rising trend of possible exploitation, but also the ease with which these exploits are accessed and accomplished.

Recent cyberattacks and malware proliferation indicate a growing sophistication of attackers and a willingness to utilize cyberwarfare to further a cause or gain a political objective. The New York Times reported on March 6, 2002 that U.S. officials had intercepted electronic communications from al Qaeda network members, and there is a growing fear that the group will utilize their "sophisticated Internet ability to launch new terror attacks against the United States."[2] The flow of information and the ability to communicate and utilize the tools of the Internet are enough of an asset that it is worth identifying the model of vulnerability detection, assessment, and response, and possible means of responding to vulnerabilities. However, in consideration of the interconnected network of critical infrastructure assets, this need becomes a national security matter. The response of system administrators to various cyberthreats varies from diligence to negligence, depending on the limitations of individual expertise and the resources available. Additionally, the initial time required to implement even the "cheap" solutions outlined in this report is still prohibitive for a staff whose main mission is to keep the infrastructure alive from day to day.

[1] Power, Richard, "2002 CSI/FBI Computer Crime and Security Survey," *Computer Security Institute,* Spring 2002.
[2] Risen, James and David Johnston, "Intercepted Al Qaeda E-Mail Is Said to Hint at Regrouping," *New York Times,* March 6, 2002.

Authors of this report:

Julie M. Cullen
Research Analyst, IRIA

Vincent Berk
Research Engineer, IRIA

Marion Bates
Research Engineer, IRIA

Input and assistance received from George Bakos, Trey Gannon, Eric Goetz, Robert Gray, Guofei Jiang, Dennis McGrath and William Stearns.

Section 1 - Release of a Super Worm

The Internet has enhanced information accessibility, and the ability to transfer ideas, conduct business, and communicate; nonetheless, this openness lends itself to vulnerabilities. The vulnerabilities that exist in networks and software have created a situation in which proprietary information and the smooth running of systems are in constant jeopardy.

The current cycle of vulnerability detection, assessment, and response:

- Identification of vulnerability
  - Inform security community and/or appropriate vendor and/or press
  - Scan for possible existing exploits of vulnerability
- Development of response, i.e. patches
  - Inform the community of security measures to be taken
  - *Hope everyone takes responsibility for applying patches*
- Observe effects of vulnerability/exploit on unprepared systems
- Learn from observations, develop new strategies and tactics

This cycle is at best inconsistent and reactive. There is no comprehensive doctrine for responding to the spectrum of threats that may emerge in the near future. In the context of this cycle, consider what might occur if a technically sophisticated worm was developed to exploit vulnerabilities in multiple operating systems. If this worm could spread before a response is developed and could be written to destroy machines affected, widespread damage would occur.

White House cybersecurity advisor Richard Clarke indicated that funding cybersecurity initiatives would not be enough to secure critical networks. Budget proposals for fiscal year 2003 have earmarked about 8 percent of the $52 billion in IT funds for security.[3] Clarke stressed it could take between three and five years of constant attention and private-sector cooperation to achieve a comfort zone with security.[4] For example, former NSA agent and Defense Department official Warren Axelrod indicated that not only is there an increase in formidable Internet-based attacks, but these attacks are also "twice as likely to affect power utilities in the United States than financial firms."[5] Directed assaults against specific financial services, electric power, and other critical infrastructures are occurring more frequently and with more sophistication.

Compounding the issue, software and hardware vendors release flawed products. Speed and optimization of profit drives the release of products and security seems to be an afterthought. Microsoft's new initiative of creating products with security as a essential feature exemplifies the problem. Serious vulnerabilities have been discovered in Microsoft products that have led to exploitation and proliferation of malware, such as the Code Red worm and the ILoveYou virus.

While resources exist to assist security professionals in their goal to guard systems against cyberattacks and cyberterror, the publication and dissemination of best practices does not in any

[3] Jackson, William, "IT security is 3 to 5 years away," *Government Computer News,* April 1, 2002.
[4] Jackson, William, "IT security is 3 to 5 years away," *Government Computer News,* April 1, 2002.
[5] Kite, Shane, "Experts: Industry Must Confront Growing Cyberthreat," *Securities Industry News*, April 1, 2002.

way guarantee that these initiatives will be implemented. In fact, a report conducted by McAfee Security of one hundred IT managers from U.K. businesses indicated that while "82 percent of respondents had suffered from a virus attack in the past 12 months (34 percent of which experienced virus-induced downtime), more than nine out of ten IT managers believed they had sufficient resources to manage security."[6] Security measures are dependent upon system administrator awareness, diligence, and deployment skills. Human error or neglect can open the door for malware proliferation and system exploitation.

Proliferation of malware is a common example of under-addressed cybersecurity vulnerabilities. Worms, viruses, and Trojan horses have become a daily event to be addressed by network administrators. In fact, the current statistics from Riptech indicate that in the second half of 2001, there were at least 128,678 security breaches verified. Had worms been in included in the tally, the number would have increased by 63 percent.[7] The estimated damage caused by viruses in 2001 according to Computer Economics researchers is an estimated $13 billion.[8]

The threat of a Superworm

The Los Angeles Times reported on April 25, 2002 that Central Intelligence Agency analysts believe Chinese military cyber-experts are developing plans to launch wide-scale cyberattacks against United States and Taiwanese computer networks. Military systems are included in the plans, and analysts believe the attacks could occur within weeks. Chinese Embassy officials, however, have asserted that their government is conducting computer research as a means of improving national security, and Chinese government policy does not include plans to disrupt computer systems of other countries.[9]

An analyst from Rand Corp. has indicated that China may conduct cyberattacks in conjunction with an invasion of Taiwan, and strike U.S. computer systems to cause disruptions to slow down U.S. intervention efforts.[10] Some analysts nonetheless believe China does not yet possess the capability to effectively carry out these cyberattacks, and likened their capabilities to sophisticated hackers capable of temporarily disrupting sectors of the Internet.[11] Furthermore, the U.S. intelligence officials believe most sensitive U.S. military networks are secure from attack, although other military networks may be vulnerable. Yet, some analysts believe that the original Klez worm variation and the Code Red worm originated in China.[12]

The threat of Chinese cyberwarfare specialists remains unconfirmed, but previous incidents provide an indication of an increased willingness of hackers from China to utilize cyberattacks to further a political agenda. Following the U.S.-China spy-plane incident, around 1,200 U.S. government and commercial Web sites were disrupted or defaced with pro-Chinese messages, and some theorize these attacks had the blessing of the Chinese government, if not direct

[6] Ahmed, Keeno, "Security Lessons from Across the Pond," *Internet World,* April 25, 2002.
[7] "Report: Hacking Is Increasing," *CBS News*, January 28, 2002.
[8] "Major viruses cost industry $13bn in 2001," *vnunet.com,* January 1, 2002.
[9] Lichtblau, Eric, "CIA Warns of Chinese Plans for Cyber-Attacks on U.S.," *Los Angeles Times,* April 25, 2002.
[10] Lichtblau, Eric, "CIA Warns of Chinese Plans for Cyber-Attacks on U.S.," *Los Angeles Times,* April 25, 2002.
[11] "China Incapable of Hacking US Files," *New York Times,* April 25, 2002.
[12] Vamosi, Robert, "Cyberwar with China? Nope--it's just May again," *ZDNet.com,* May 1, 2002.

involvement.[13] In April 2001, the anniversary of the U.S.-China spy-plane incident passed without an obviously connected incident, but May 8, 2002 is the anniversary of the 1999 U.S. accidental bombing of the Chinese Embassy in Belgrade.

The fundamental questions examined in the following passages are: Can a worst possible case scenario be identified, for example the SuperWorm, that would threaten the economy or national security enough to warrant suspension of all Internet activity? What are possible features of the SuperWorm that could cause widespread damage? What is the current environment in which this worm would be released? What can be done to proactively defend against it, what can be done to detect it early, and what are the possible responses once it is out there in the wild? While it seems highly unlikely that such a set of circumstances would come to pass, few anticipated the full shut down of air traffic after September 11, 2001. The result of stopping all air traffic into and within the United States had a massive effect on the tourism and shipping industries, the financial sector, and other sectors or industries dependent on air traffic. If a worm was identified that could destroy any machine affected, the impact of destroyed machines or pulling machines from the Internet would be severe for the financial sector and many other sectors.

Finally, if such a worm were created as a means to further political agenda, similar to the threat identified by CIA analysts or the threat of cyberwarfare by members of the al Qaeda network, significantly disrupting or destroying the communications infrastructure could have extensive political ramifications.

The features of a SuperWorm that would cause widespread damage

Self-propagating malicious programs moving amongst interconnected computer systems have come to be commonly referred to as worms. Although there are a large number and types of malicious programs, completely autonomous worms, infecting and replicating without requiring any action on the part of the system operator, are potentially the most dangerous. Additionally, recent worms have exhibited explosive growth rates that far outpace efforts to stem their activity. CodeRed (CR) and CodeRed2 (CRv2) took advantage not only of programming flaws in Microsoft Internet Information Services (IIS), but also of default installation settings. In order to facilitate rapid propagation, CR used a random number generator process as its target selection algorithm. Initially, the random number generator suffered from a static seed, thus every copy of the worm scanned the same IP space over and over. A modified version of CR was released shortly thereafter to use host time as a pseudo-random seed. With this enhancement, more than 359,000 computers were infected in under 12 hours.[14] The payload was mild, however: a simple time-delayed Distributed Denial of Service (DDOS) attack against whitehouse.gov's IP address. The DDOS attack was targeted at the IP address of whitehouse.gov, and a countermeasure was quickly developed and the threat mitigated by simply changing the IP address. The Nimda worm followed the Code Red worms, making use of the same IIS vulnerability but employing a number of other exploit mechanisms to increase its propagation capability.

[13] Lichtblau, Eric, "CIA Warns of Chinese Plans for Cyber-Attacks on U.S.," *Los Angeles Times,* April 25, 2002.
[14] Moore, David, G.M. Voelker, and S. Savage, "Inferring Internet Denial-of-Service Activity," 2001, accessed from http://www.cs.ucsd.edu/~savage/papers/UsenixSec01.pdf

As the CodeRed and Nimda worm examples indicate, those intent on the proliferation of malware can learn from the flaws and successes of the worm to generate a more dangerous worm. While Code Red and Nimda made headlines with benign payloads, several 2001 Win32 viruses, including the W32/magistr worm, did not receive as much attention but contained extremely destructive payloads. What if the propagating speed of a Nimda-like worm was combined with the destructive power of magistr, which overwrites hard drives, erases CMOS settings, and flashes the BIOS chip when activated? If the most damaging aspects of known and theoretical worms were combined, the effect would be the creation of a SuperWorm.

Nicholas Weaver of the University of California, Berkeley theorized that the development of a SuoerWorm is possible:

> "The reality is frightening: a couple of skilled programmers, with ownership of a small number of isolated machines for testing, could easily create the framework [for a SuperWorm] in a couple of months. When a new, significant exploit is discovered, either by the authors or someone else, the worm could be quickly adapted and released into the wild. Even after release, new exploits and routines could be easily added, to increase potential disruption."[15]

The theorized SuperWorm would be able to spread quickly, damaging as many systems as possible. In reality, a combination of the features of known worms that have been the most successful would create a serious threat: the speed of Code Red and CodeRed2, the multiple means of proliferation utilized by Nimda, the ability to incorporate other virus code to be transported with the worm seen with variations of the Klez worm, the ability to access and send files from a hard drive as seen with SirCam, and the destructive payload of Magistr.

The SuperWorm has three tasks:

- Exploitation of various vulnerabilities that affect a large number of operating systems
- Propagation
- Delivery of payload and damage

Exploits vulnerabilities that affect a large number of operating systems, and exploits multiple vulnerabilities

The target selection algorithm of SuperWorm will contain a set of instructions that indicate the manner in which the worm will probe the Internet and exploit vulnerabilities. The SuperWorm could contain compiled code that can affect both Unix and Windows systems. Each feature of the worm will work in conjunction with others; in this case, to damage the maximum number of systems the attacker will trade off virulence for speed. For example, the speed by which the worm is able to spread may be slowed by the worm's methodology to locate new targets.

[15] Weaver, Nicholas, "The Capability to Create a SuperWorm," accessed from http://download.demonics.org/papers/capability.txt.

SuperWorm can originally be programmed to exploit set number of vulnerabilities. For example, if five vulnerabilities have been identified, the worm will attempt to affect a system by attempting each vulnerability:

Attempt to exploit vulnerability A
Then B
Then C
Then D
Then E

The SuperWorm will exploit on a vulnerability curve based on the commonness of the vulnerability, or based on the severity of a compromise, and will attempt to infect the machine with each of the known vulnerabilities until the system has been compromised. The current trend of malware proliferation begins as an attacker identifies vulnerabilities in operating systems. For example, Code Red and Nimda spread through the IIS vulnerability. These worms took advantage of a canonicalization error in the popular Microsoft IIS 4.0 and 5.0 web server that allowed a malicious user who visited the web site to gain additional privileges on the machine. Nimda also changed the main page on the machine, such that visitors to the site would also become infected by the worm. Nimda did not deface the page in any appreciable way, but inserted a java script that would ask visitors if they wanted to download a 'read me' file. If the browser offered a dialogue box, the user had the choice to download the innocent sounding file; some browsers automatically downloaded the malicious code. Once the user downloaded the malware, the worm would acquire addresses from the user's e-mail program in order to e-mail itself to other machines. In this manner, the Nimda worm exploited multiple infection vectors.

The most lethal worm would be able to spread through multiple processes. Additionally, the worm could exploit multiple operating systems by containing code capable of infecting any potential operating system. For example, Windows systems with vulnerability A will be infected as will Linux systems with vulnerability B, by including the vulnerabilities found for multiple platforms into the infection algorithm. For example, CERT®/CC reported in May 2001 that the sadmind/IIS Worm affect both Microsoft and Solaris systems. During the probe phase, the worm will identify vulnerable systems to attack.

Propagation and multiple means of proliferation

There are strategies of propagation and improving propagation that could be beneficial to the SuperWorm: utilizing a **central source** to collect and analyze data gathered by the worm to create more effective versions of the worm, or **back-chaining** to optimize the speed of the worm.

The worm could have the ability to send back data to a central source, enabling the attacker to update the code. Subsequent to release, the worm will continue to propagate through its mass-mailing capabilities. As the worm compromises systems or machines it could send information back to a central source, such as a designated e-mail account. Weaver believes that a "worm which is modular, communicates and coordinates with other running copies, and is able to accept and distribute cryptographically signed modules (implemented in a scripting language and as loadable libraries) would be an excellent platform for information warfare and a significant advance over current worms."[16] More efficient or effective target selection and vulnerability exploits can be learned from data gathered during earlier scans and worm proliferation attempts. The attacker can release new versions of the worm with the updated code.

There are two additional central source methods that can be utilized to enable communication from the SuperWorms to the attacker and from the attacker to the SuperWorms. News group postings and Internet relay chat communications could allow two-way communications.

The SuperWorm could be coded to report information to specific news groups. The attacker could program the worms to communicate with a particular signature or key that can easily be identified by the attacker during a search of the postings to the news group. This allows the attacker to detect messages posted by the worms. Additionally, the worms could be programmed to check back to the specific news group postings to access further instructions, or code updates, from the attacker; again, the attacker's communications could have a recognizable signature that the worms can be programmed to identify. The communications could be encrypted or hidden within large documents to avoid detection.

Internet relay chat communications could not only be the information posting location for the attacker to access, but will also inform the attacker which machines are affected by the worm. Weaver indicates that the SuperWorm could be programmed to determine whether or not it is able to accept incoming messages on a particular infected machine, and if so, send the IP address of the machine to the central source(s).[17] Again, the worm could be programmed to communicate through an IRC, and the attacker can monitor the chat room and still maintain some level of anonymity by changing machines or utilizing encrypted communications.

The serious point of weakness in utilizing a central source is that the location, or locations, of the site to which packets are returned can be discovered and shut down. Ultimately, use of the central source will both slow down the worm, as communication is time-consuming, and is vulnerable to detection.

Back-chaining is a fully autonomous method of proliferation wherein the worm makes an exact copy of the code to send itself to the next machine, regardless of current source or location. Examples of this method of proliferation were seen in the Code Red and Nimda worms. This method does not rely on a central source, but propagates from machine to machine, without the ability to utilize a central source to update the code. However, one way to update the number and type of vulnerabilities contained in the code would be to install a back door in the infected machines and communicate through a central source.

If the best features of each method are combined, the SuperWorm could be a polymorphic worm that changes its source code during each propagation. For example, the first copy of the worm has targeted the universe of all possible IP addresses. An algorithm could be created that would constantly update the IP database used by the worm to ignore the IP address of the machine now being infected. Another possible algorithm can be found in **Figure 3**. Therefore, the need for

[16] Weaver, Nicholas, "The Capability to Create a Superworm," accessed from http://download.demonics.org/papers/capability.txt.
[17] ibid.

central source is now unnecessary. The algorithm will reduce redundancy by eliminating the possibility that the worm would try to re-infect machines that already contain a copy of the code. The SuperWorm could be developed to spread without human intervention. If this worm exploits vulnerabilities and is not dependent upon any human behavior for infection, then the worm could not only spread more quickly, but it could also infect machines of even the most vigilant and aware users.

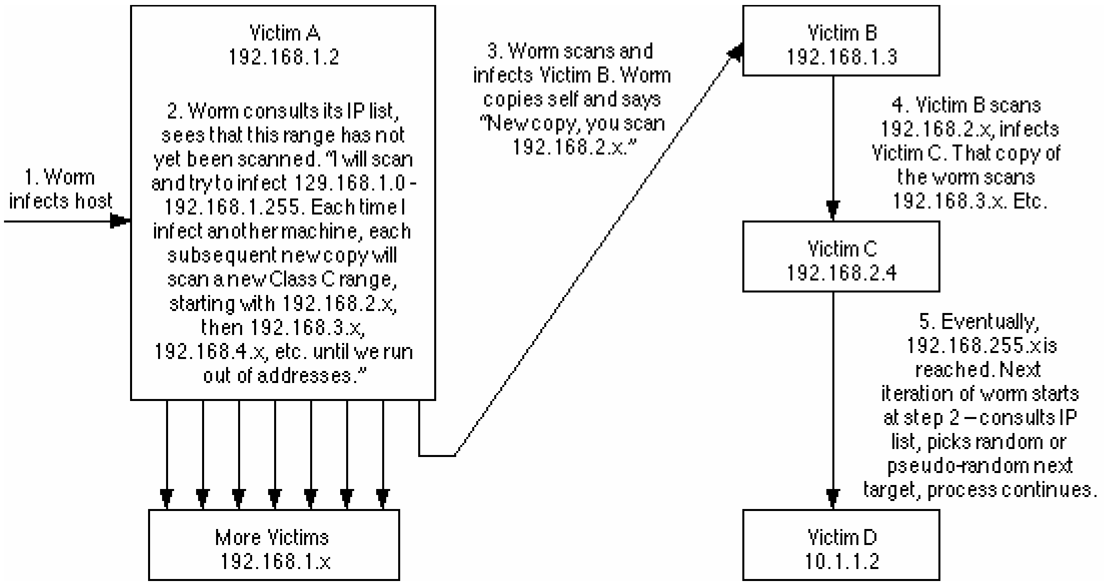
Victim A
192.168.1.2
2. Worm consults its IP list, sees that this range has not yet been scanned. "I will scan and try to infect 129.168.1.0 - 192.168.1.255. Each time I infect another machine, each subsequent new copy will scan a new Class C range, starting with 192.168.2.x, then 192.168.3.x, 192.168.4.x, etc. until we run out of addresses."
1. Worm infects host
3. Worm scans and infects Victim B. Worm copies self and says "New copy, you scan 192.168.2.x."
Victim B
192.168.1.3
4. Victim B scans 192.168.2.x, infects Victim C. That copy of the worm scans 192.168.3.x. Etc.
Victim C
192.168.2.4
5. Eventually, 192.168.255.x is reached. Next iteration of worm starts at step 2 – consults IP list, picks random or pseudo-random next target, process continues.
More Victims
192.168.1.x
Victim D
10.1.1.2

FIGURE 3

The theoretical worm, affecting multiple identified vulnerabilities in multiple operating systems, would need to spread quickly, before the security community is alerted and responds to the vulnerabilities. Other known worms can serve as the model of successes and failures. For example, Code Red attacked servers on network-connected machines. Its speed was determined by the efficiency with which it located new targets. Code Red was not as efficient as it could have been as it scanned indiscriminately, probing non-vulnerable machines by scanning random addresses. Additionally, it tried to infect machines running other copies of the worm. Nonetheless, scanning itself adds to the congestion of the network traffic, and decreases the availability of resources on the Internet. The SuperWorm could be programmed to utilize an automated scanning tool and to locate and identify only those machines that can be infected. When launched, the worm would only target vulnerable machines, giving the security community little time to detect and identify a counter-measure to mitigate the attack. There are a number of known automated port scanning tools that are readily available for download and use.

Worms typically have taken hours to spread throughout the world. If the creator of the SuperWorm utilizes the ideas of the theoretical "Warhol Worm" developed by researchers at Berkley, the worm might be capable of spreading to all vulnerable hosts within 15 minutes.[18] This 'super virulent' worm could utilize optimized scanning routines that are capable of scanning

[18] Weaver, Nicholas, "Warhol Worms: The Potential for Very Fast Internet Plagues," February 13, 2002, accessed from http://www.cs.berkeley.edu/~nweaver/warhol.html.

without using significant bandwidth, minimizing the chance of setting off alerts. If hitlist scanning is conducted, a list of targets is developed that identifies vulnerable machines. As counter-threat measures are being developed, the worm could be spreading to all vulnerable machines.

Delivery of payload and damage

In this final phase, the SuperWorm will settle into the machine and deliver its payload, hide from detection while this is being accomplished, and continue its propagation.

Bypass and Disable Anti-Virus Software:

Anti-virus software can only detect known signatures, and if the SuperWorm varies enough from known signatures, it may pass undetected. It would be unnecessary to disable security measures if the intent of the worm is to deliver malicious payload. Disabling the anti-virus software only benefits the attacker if serious damage is intended for only certain machines, or if the attacker wishes to gain access to affected machines. For example, the attacker may wish to destroy machines with IP addresses registered to .com and .edu, but intends to gain privileges on other machines, such as those registered to .gov or .mil. If the worm could detect which type of machine it infected, it might be programmed to copy and send files of the machine to a remote user. SirCam was able to attach files from the hard drive and e-mail them with a copy of the infected virus. The loss of critical or even sensitive information to the attackers could create a much more serious situation than simple virus infection. Klez.E and Goner are examples of worms that attempted to disable or disrupt anti-virus software or firewalls.

Delivers a Malicious Payload

The malicious payload is the biggest threat from the SuperWorm. The polymorphic Magistr Worm spread through a vulnerability in Microsoft Outlook and contained its own SMTP code so that it could exploit even non-Microsoft Outlook users. When downloaded, Magistr destroyed contents of the hard drive. The significance was the destructive payload: Magistr will "destroy the computer's CMOS/BIOS information as well as sectors on the hard drive on Windows 95, 98, and Me systems. Magistr will overwrite every 25th file with the words YOUARESHIT as many times as it can, and delete every other file on the hard drive."[19] If the SuperWorm had a similar malicious payload, and was capable of spreading to multiple operating systems rapidly, the destruction would be severe.

Possible results of a SuperWorm

If this combination of features were synthesized into one worm, the effects could be devastating. The goal of the worm would be to spread immediately, without human intervention, and before counter-measures are developed, to exploit vulnerabilities of widely-used operating systems and deliver a malicious payload. Additionally, it is not out of the realm of possibility to think that a

[19] Vamosi, Robert, "Don't get sentenced by the Magistr virus," *ZDNet Reviews.* April 1, 2001.

terrorist group or another state would spread such a worm in conjunction with, or prior to a physical attack.

Finally, if such a worm were created and intended to destroy all compromised systems, it would be unnecessary to cover tracks or hide the source of the attacker. Additionally, if SuperWorm was released prior to the development of anti-virus measures to detect and remove the code, the worm would not need to be self-encrypting or utilize polymorphic measures to defeat detection. The worst-case scenario would be to render machines hit by the SuperWorm inoperable, or access and send valuable information from an effected system to the attacker.

The SuperWorm raises questions of the means by which security experts can protect their systems against infection. In reality, system administrators have limited resources to keep up with the patches necessary to address vulnerabilities of each networked computer, and vulnerabilities are increasing not decreasing. It is feasible, however, that we could improve detection and response times. A careful and realistic examination of the features of a computing environment that could lead to vulnerabilities will provide systems administrators with proactive steps that can be taken to enhance network security. The incident at XYZ University provides an example of an environment in which vulnerabilities tend to be addressed after an incident occurs.

Early detection of worm traffic and scanning activities may provide cybersecurity experts with enough time to address vulnerabilities and inoculate systems. The section on Early Detection of Active Internet Worms by Metering ICMP Destination Unreachable Messages will provide an example of a method by which an early warning system can be established. However, detection of a worm does not ensure that it will not cause damage. By July 19, 2001, it was estimated that Code Red and Code Red2 had compromised more than 15,000 machines.[20] Months later, on May 3, 2002, an analyst from Arbor Networks indicated that the threat from Code Red still exists as some computers have not been patched to address the vulnerability exploited by Code Red.[21] Education and a means to alert and convince users to patch their systems are essential. Unfortunately, it seems that constant publication of warnings may desensitize users to threats.

A final security technology to be examined will introduce a tool that would mitigate threats from worms to web servers. The modified reverse proxy server (JEANNE) is designed to eliminate the possibility of malicious traffic passing through the firewall into the network.

[20] Lemos, Robert, "Code Red Worm Set to Flood Internet," *C-Net News.com,* July 19, 2001.
[21] Lemos, Robert, "Code Red Still Threatens Net," *C-Net News.com,* May 3, 2002.

Section 2 - Incident at XYZ University

The incident at XYZ University provides an actual example of how network security professionals respond to cyberattacks. The cycle of vulnerability detection, assessment, and response is examined in the context of an actual security incident that occurred in a university setting. The nature of the network, relatively open and accessible to promote knowledge and information sharing, and the resources that those responding to the incident had at their disposal combined to create a situation in which the vulnerabilities of one machine could be exploited and used to attack other machines. The security of this machine was easily compromised, but the attacker drew attention to the compromised machine through his/her activities, and system administrators were able to address the problem. The incident itself indicates one current environment in which the SuperWorm would have little trouble spreading. The security analyst's examination of the conditions that helped facilitate the exploitation of a vulnerability provides insight into the means by which cybersecurity professionals respond to an incident.

The security analyst examining the incident examined the features of the network and machine that created the vulnerability, the means by which system administrators were made aware of the problem, the response of system administrators, and general lessons learned from the incident.

When a cyberattack or other incident occurs, security professionals will often contact the IP source of the attack, and ask for an explanation or request the owner of the particular server or Internet Service Provider to stop the attacker. Cooperation is variable, and the outcome, especially if it involves an attack from across international lines, is unpredictable.[22] In order to locate the attacker and to preserve the evidence of the attack, initial actions taken by systems administrators to address an incident should be done with the consideration of not tipping off an attacker who may control the system. This will not only aid law enforcement officials in their pursuit and successful prosecution of the cyberattacker, but will also provide information that others can analyze and learn from to prevent future attacks. Speed in determining the nature and source of the exploit is vital to stopping an attack. The following incident report will help illuminate some problems facing security professionals, and factors contributing to the vulnerability of the current heuristic for contending with network attacks.

In 2002, a student's computer at XYZ University was performing portscans against a number of networks around the world. [Due to privacy concerns, indicators have been removed from the report.] The exploit used might have been an early variant of the Ramen worm, but due to incomplete forensic evidence, it is difficult to conclusively determine.

The analyst considered the following threat vectors to determine the source of the attack:

- Outsider attack from a network
- Outsider attack from a telephone
- Insider attack from a local network
- Insider attack from a local system
- Attack from malicious code

[22] Kite, Shane, "Experts: Industry Must Confront Growing Cyberthreat," *Securities Industry News,* April 1, 2002.

Based on the fact that the machine was within a large, relatively insecure network, and the logs retrieved from the student's machine showed evidence that a non-University address had gained privileges on the machine, the analyst deduced that the attack was a remote exploit, an outsider attack from a network. The other possibility was an insider attack from a local system. The elements of the attack led the analyst to determine that the machine had most likely been compromised remotely, and was being used by the attacker to port scan other IP addresses.

Security Conditions of XYZ's Network

The environment is a university campus network with around 6000 hosts, running an assortment of operating systems: 30-40 percent Windows98/NT/2000/Me, approximately 40 percent MacOS, and the remaining run various flavors of Unix and Linux. The particular machine in this case was a student's personal machine (located in a dorm room) running a default installation of RedHat Linux 6.2. The student told computing staff that he wanted to learn Linux, and chose RedHat 6.2 because it was stable, fairly current, and easily installed.

The university's network maintains some firewalling/filtering, but it is not extensive due to the requirements of users and the large scale of the environment relative to the size of the support staff. Some traffic is logged, but due to volume, the logs are not archived: they are purged after two or three days. Log analysis can provide early warning of malware proliferation. More details of this process will be provided in the section on Early Detection of Active Internet Worms by Metering ICMP Destination Unreachable Messages. It is unknown whether or not there is methodical log analysis taking place on a regular basis, but the analyst performing the review believes that attention is focused only when there are noticeable aberrations, as in the case of this incident, limiting the ability of system administrators to observe early worm traffic.

At the time of the incident, the university's border router was configured to block and log directed broadcasts to all campus subnets, to filter traffic with bogus source IPs (i.e., incoming traffic with source IPs within the university's IP space and private network addresses such as 192.168.x.x), and to block SNMP traffic from all but a couple of trusted IPs (presumably for router management or statistics). All other traffic destined for their IP space was allowed. Due to the rules and their ordering, it would have been possible for some bad traffic to slip by without being logged. This is understandable, as more filtering would be liable to break "normal" functionality, and many campus departments share information with their academic, government, and public partners over a variety of server protocols. Therefore, nearly all ports must be open at all times to avoid complaints and headaches.

Beyond the border router, there is little or no filtering in place, according to campus system administrators. Some separate departments have their own firewalls, such as the Computer Science department, but student dorms are connected directly to the campus backbone, and no additional firewalling is performed within the network. **Figure 4** provides a simplified network diagram – border router, upstream providers, and campus backbone. Information was furnished courtesy of the university's internal network mapping program.

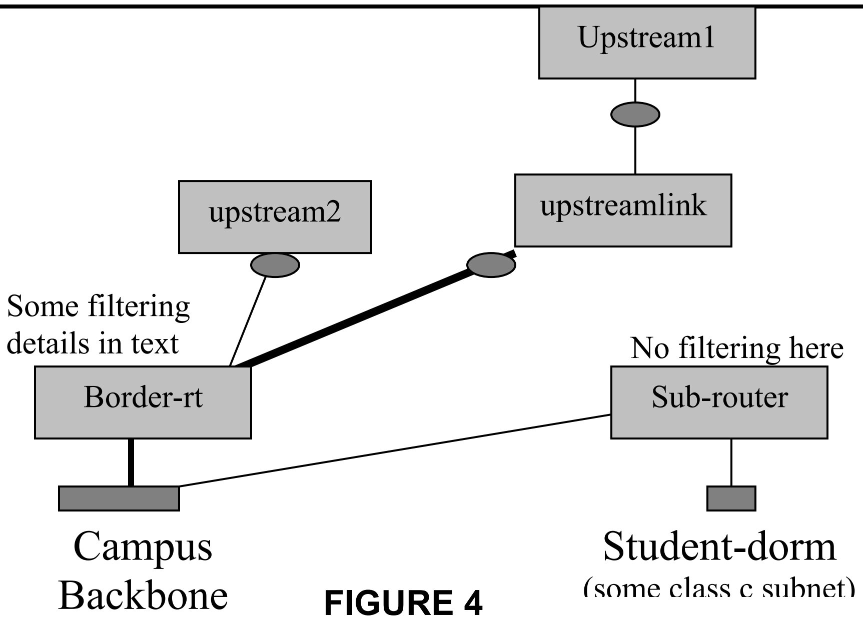

FIGURE 4

The university's security staff maintains a separate e-mail address for network abuse, "abuse@xyzuniversity.edu", and an internal mailing list for the use of university security staff and individual department system administrators to discuss security issues and report break-in attempts on their respective portions of the network. Any university staff member responsible for any portion(s) of the university network can subscribe to this list. Posts range from queries for advice ("I have a machine that appears to be infected with Nimda, what do I do next") to general announcements ("SomeBigCompany just released the 37th patch to their web browser"). Some departmental sysadmins are held responsible for maintaining security on research or public lab machines, but sysadmin do not "patrol" individual student machines in dormitories.

The biggest wildcard on the network is the unfettered presence of student machines running any OS with any server configuration a student prefers. On the positive side, students are allowed and even encouraged to educate themselves about new software programs and operating systems in their own dorm rooms, but the lax policy creates a logistical issue for campus network administrators. Due to the number of machines on the network, it would be extremely difficult to take proactive measures to ensure the security of each machine on the network. Time and resources do not permit the university's security staff to examine individual machine security.

It is unknown at this time whether or not the support staff has a handbook or equivalent "official" set of guidelines to follow when handling incidents of this nature.

Description of the incident

In late January and early February 2001, the university's network-abuse e-mail account received messages from organizations around the world reporting portscans against their networks. The scans originated from an IP address within the university's network.

Each of the messages detailed the type of attack on their system, and requested that the university staff examine and resolve the issue. One significant trend identified from this series of events is the tendency for a sysadmin who has noticed the portscans to identify the apparent source of the scans and directly contact the sysadmins of the network from which the attacks originated. As indicated earlier, it can be a significant challenge to unravel the actual source of the attacks, as will become clear with the examination of this incident. There is no indication that law enforcement officials were alerted to the incident.

The overwhelming majority of the reported attacks were probes against port 111, which is used by the SunRPC Portmapper service. (RPC stands for Remote Procedure Call -- the RPC services provide means for remote execution of programs on the server.) Additionally, there were reports of traffic to port 21 (ftp). The analyst assumed that perhaps there were more port 21 probes, but that they went unreported due to the greater volume of FTP traffic in general -- perhaps these were lost in the "noise" of legitimate traffic or misconfigured intrusion detection systems. The File Transfer Protocol port has historically been a big target due to the large number of vulnerabilities discovered in the standard FTP daemon (wu-ftpd).

The volume and pattern of the probes suggests the use of an automated tool that scans entire ranges of IP addresses. It could be a worm, or maybe a reconnaissance tool used by an attacker to find other vulnerable systems. It is unlikely that the attacker performed the scans manually, though it is possible. **It is most likely that the machine was hijacked and utilized by a remote operator to perform portscans of networks around the world. While it appears that the attacks originate at XYZ University, the real source of the attack remains undetermined as forensic evidence was either not collected or not adequately preserved for analysis.** This incident illuminates another tool for use by the SuperWorm's creator. If the attacker in this incident were port scanning IP addresses world wide to develop a hit list of machines vulnerable to a particular exploit, SuperWorms could be quickly sent to exploit machines identified in the scans.

Action taken by Sysadmins at XYZ University

The university's network security personnel issued a reply to those whom had identified the incident and notified the university. The reply indicated that the machine would be disconnected from the network until it could be cleaned. Additionally, following message was sent internally to university security staff:

```
Date: Fri, 02 Feb 2001 08:30:41 –0500
From: bob ------- <bob@XYZ.EDU>
Sender: security-list@XYZ.EDU
Precedence: special-delivery
Reply-To: security-list@XYZ.EDU

Folks,
This attack came from a student computer named tatooine.XYZ.edu. Last week we had
to block him from the net. This week he said he had cleaned up the machine, so I
unblocked him. Either he lied to me or he is not competent to do the job. I'll reblock him
and talk to him.
bob
```

The referenced prior block placed on the student's access was due to earlier reports of this same sort of malicious network activity. There was little information left for the analyst to examine from the "original" incident. This largely indicates that the computing staff was more inclined to do damage control the first time, but did a more in-depth investigation after subsequent occurrences. **It is possible that the second incident could have been avoided, had the first incident been more thoroughly investigated and the follow-up been more closely scrutinized.** It was damaging enough that the proactive security measures of the university were insufficient to stop the first attack, but the incident was not adequately addressed to stop subsequent attacks.

Eradication and Recovery

The following recommendations could have helped secure the Linux box prior to exploitation. However, even the development of a comprehensive security plan does not ensure that individuals will secure their systems. This section examines some security measures that can be taken and some of the user errors and misconceptions that limit their effectiveness.

One security team member was dispatched to explain to the student what he needed to do to prevent a recurrence of this event:

> "From memory, the instruction I gave the student was:
> - run up2date to install all avail patches from priority.redhat.com
> - turn off most things in inetd, turn off most things in the runlevel rc.
> - check the university linux web page
> - run the bastille hardening script"

The up2date utility is part of the Red Hat Network. Once configured, it automatically checks for updates and patches, and can download and install needed software by itself, making the upgrade process much easier to manage. Automation of the update and patch process is of critical importance in an environment where users are of varied skill levels and the security support staff has limited resources.

Inetd is the Linux "superserver" which handles startup for a number of well-known network services/daemons, such as ftpd, telnetd, fingerd, chargen, etc. Most of these are seldom needed, especially on a single-user machine; however, on a stock RedHat 6.2 install, many of these are enabled by default. Installation of many of these services can lead to vulnerabilities due to user error and lack of technical sophistication.

The university's webpage on Linux security discusses some general security ideas, such as using strong passwords (examples are provided) and creating non-root-user accounts. It also provides a fair amount of detail about disabling services in inetd.conf and rc.d and checking the system status with netstat. The university also provides a Linux users mailing list, which is monitored and maintained by a handful of Linux/Unix experts. In addition, users are encouraged to check other sources of information such as the comp.os.linux.security FAQ and to make sure their systems are running the latest patches. **Installing security patches in this case would have prevented the attack**.

Due to the complexity of the task, this can be daunting for a new user. Many Linux services are automatically configured to launch when the machine boots up. In many cases, it is easy for an inexperienced user to temporarily disable some services and therefore believe that the machine is secure, only to discover later that all those services reappeared after a reboot. Disabling startup programs under Linux is in many ways a non-intuitive process. Even determining which services are network-related can be challenging, and many users are afraid of crippling a fresh Linux install by turning off too many of them. This, coupled with the widespread belief that "well, if I have nothing valuable on this computer, then no one will bother to try to break into it" leads to an abundance of misconfigured/unsecured machines exposed to the Internet.

The webpage also recommends use of the Bastille hardening script, which "enhances the security of a Linux box, by configuring daemons, system settings and firewalling." (sourceforge.net summary) The script uses a question-and-answer format (with detailed explanations at each step) to determine the user's needs and can automatically secure the box accordingly.

University security staff told the student that once this was done, they would re-enable his Internet access. After a few days, he announced that he had followed their instructions to make the box secure, and his access was restored. However, within a day or two, the ports canning reports began coming in again. His access (via the manually-assigned IP he requested) was blocked once again and remained blocked pending verification of the machine's security.

Even after the university disabled the student's Internet access (most likely achieved by modifying the configuration of that building cluster's router), he still had the ability to use the campus network by switching over to DHCP (the standard method for campus computer users to connect to the network). In fact, he did exactly that, in order to use the Windows side of his machine so he could continue doing schoolwork; he refrained from connecting the network while he worked on securing the Linux partition of that box.

The university's policy of blocking his access based on the static IP he had requested can be easily circumvented. As it is currently implemented, this is an "on your honor" policy -- there is nothing in place to prevent him from putting a vulnerable box on the DHCP network, which would make it more difficult for the staff to track it down. After blocking the problem machine's Internet access the second time, university security staff began a more detailed analysis. Using nmap, they first portscanned the student's machine. The results indicate a number of ports were open on the student's machine. The key open port in this case is most likely the SunRPC Portmapper port (TCP/111). RPC scans of this nature were seen to increase dramatically around the beginning of that year, and older, vulnerable versions of rpc.statd and wu-ftpd were the services that the Ramen worm and its variants exploited in order to gain root access to their victims' machines.

As is the case with most Internet worms, Ramen-infected systems are configured to scan ranges of IP addresses (sometimes random lists of IPs, sometimes IPs on the same subnet as the victim machine, sometimes even hard-coded specific targets). If other vulnerable systems are found during these scans, then they too become infected with the worm, and the process repeats itself. The Lion worm, which is more malicious, is a variant of Ramen.

The limited information available prohibits conclusive determination of whether or not this was an instance of the Ramen or Lion worm (or another early variant), or whether it was simply a similar attack method taking advantage of the same vulnerabilities. The fact that the targeted ports were 111 and 21, that the IP ranges scanned were close together numerically, and that the t0rn rootkit was used, all suggest that this incident could very well have been caused by a worm such as Ramen or Lion. If true, this tells us that the worm (or an earlier version of it) was "in the wild" a full month before the earliest official reports.

After the sysadmins' reports came in, the same member of the security team was sent to examine the box in person:

> "What I discovered on the student's machine: There were two hidden processes, several binaries were most likely trojaned, I searched for a root kit but was unable to locate it. The logfiles were clean as would be expected with a decent linux rootkit. The Ethernet interface was not in promiscuous mode, so there didn't seem to be evidence of an intruder's sniffer program. However I wanted to put together some clean binaries of tools and dig around some more but limited time to explore and the student's decision to reformat/reinstall eliminated the need."

After the second "wave" of incident reports, a university computing staff member used chkrootkit to scan the student's machine and generate a report. This tool is a shell script that examines the binaries (compiled system files) for modifications that might indicate the presence of a rootkit or other compromise.

Chkrootkit did not detect Ramen/Lion-related files, which contradicts the analyst's earlier hypothesis that this might have been an early instance of one of those worms. It is still possible, however, that the attacker manually removed the files beforehand, or that the files were altered to evade detection by chkrootkit, possibly indicating an unrecognized variant of the worm. It is also possible that this check was run against the filesystem some time after the event, using an updated version of chkrootkit which included the ability to detect the (now known) Lion and Ramen worms. Also, the student might have altered the filesystem prior to this analysis, thus misleading chkrootkit.

Chkrootkit did report "Warning: Possible LKM Trojan installed" based on the two hidden processes it detected. LKMs are Loadable Kernel Modules, and trojans of this type (e.g., knark, adore) can reside in memory and can evade many detection methods. They can disguise their presence and the presence of other malicious files and processes by intercepting commands and their results; they do not need to replace binaries with trojan versions, instead they simply change the output of commands that would reveal their presence on the victim machine. This prevents the user from using tools like Tripwire to discover the compromise -- Tripwire will report that the system binaries are still valid, because they have not been changed.

According to some mailing list archives, newer versions of the t0rn rootkit include an LKM. It is possible that the hidden processes discovered by chkrootkit were more symptoms of such a t0rn variant. Without more information about the hidden processes themselves, it is impossible to determine exactly what their presence indicates with regard to this incident.

The analyst did find some interesting leftover log data, perhaps overlooked by the attacker's log-erasing tools (or the student's access was blocked before the attacker could erase them all). The victim sent these to the analyst long after the second compromise, while the student's Internet access was still blocked by the computing staff. Here is an excerpt from /var/log/messages, on the same day as one of the reported batches of portscans:

```
Jan 27 00:12:42 tatooine syslogd 1.3-3: restart.
Jan 27 00:13:05 tatooine syslogd 1.3-3: restart.
Jan 27 00:39:30 tatooine inetd[487]: pid 1741: exit status 1
Jan 27 00:40:17 tatooine syslogd 1.3-3: restart.
Jan 27 00:40:34 tatooine syslogd 1.3-3: restart.
Jan 27 00:48:15 tatooine kernel: 216.27.48.198 sent an invalid ICMP error to a broadcast.
Jan 27 00:48:25 tatooine kernel: 216.27.64.18 sent an invalid ICMP error to a broadcast.
Jan 27 01:29:46 tatooine inetd[487]: pid 2518: exit status 1
Jan 27 03:58:35 tatooine telnetd[4572]: ttloop: peer died: EOF
Jan 27 03:58:35 tatooine inetd[487]: pid 4572: exit status 1
Jan 27 04:02:01 tatooine anacron[4603]: Updated timestamp for job `cron.daily' to 2001-
01-27
Jan 27 12:09:48 tatooine syslogd 1.3-3: restart.
Jan 27 12:33:51 tatooine kernel: t0rns uses obsolete (PF_INET,SOCK_PACKET)
Jan 27 12:33:51 tatooine kernel: eth0: Setting promiscuous mode.
Jan 27 12:33:51 tatooine kernel: device eth0 entered promiscuous mode
Jan 27 12:33:59 tatooine kernel: eth0: Setting promiscuous mode.
Jan 27 12:34:26 tatooine inetd[487]: pid 5403: exit status 1
Jan 27 13:35:47 tatooine PAM_pwdb[5872]: (login) session opened for user tatooine by
(uid=0)
Jan 27 13:48:09 tatooine inetd[487]: pid 5871: exit status 1
Jan 27 17:55:02 tatooine PAM_pwdb[5968]: check pass; user unknown
Jan 27 17:55:03 tatooine login[5968]: FAILED LOGIN 1 FROM ppp3144.dragonbbs.com
FOR t0rn, User not known to the underlying authentication module
Jan 27 17:55:11 tatooine inetd[487]: pid 5967: exit status 1
Jan 27 18:00:16 tatooine syslogd 1.3-3: restart.
Jan 27 18:19:49 tatooine inetd[487]: pid 6368: exit status 1
Jan 27 18:20:54 tatooine rlogind[6376]: Connection from 38.37.0.237 on illegal port
Jan 27 18:20:54 tatooine inetd[487]: pid 6376: exit status 1
Jan 27 18:20:57 tatooine gnome-session: [gnome-session] connect from 38.37.0.237
Jan 27 18:22:03 tatooine telnetd[6380]: ttloop: read: Connection reset by peer
Jan 27 18:22:03 tatooine inetd[487]: pid 6380: exit status 1
Jan 27 18:22:03 tatooine inetd[487]: pid 6373: exit status 1
Jan 27 18:22:03 tatooine telnetd[6384]: ttloop: read: Connection reset by peer
Jan 27 18:22:03 tatooine inetd[487]: pid 6384: exit status 1
Jan 27 19:47:29 tatooine inetd[487]: pid 5969: exit status 1
```

This log excerpt indicates that at some point around the time that the scans took place, there was some suspicious activity taking place on this box. First, we see that syslogd (the Unix system logging utility daemon) is restarted five or six times within an hour, which is abnormal. (Typically, syslogd is restarted once a day; but even if it is configured to reboot more frequently, the time intervals between restarts should be regular.) We can guess that this syslog abnormality was caused by the attacker's rootkit; many rootkits include log-erasing or altering utilities, or even a trojaned version of the syslog binary. Perhaps the attacker did not know enough about how to use the rootkit tools, or a rootkit tool malfunctioned, because the logs still indicated foul play (although there was a lot of missing information elsewhere, so perhaps the logs were only altered before or after this point). We also see a couple of messages regarding ICMP errors; there is not really enough information for the analyst to determine whether the ICMP errors were related to the incident, although some rootkit utilities include hidden services which can be "triggered" (enabled) by customized ICMP packets, thus providing an obscure back door for the attacker to visit later.

Inetd exit occurs several times within the same few hours. Perhaps some inetd services were replaced by trojaned versions, or reconfigured to give the attacker back door access. Services like ftpd and telnetd are launched by inetd, and an attacker with root access will often set up telnetd (or a trojan version thereof) to listen on a nonstandard/upper port, with root shell access attached to that service. This modification is trivial to achieve if an attacker has root access; they need only add a line or two to the inetd.conf file and restart the inet daemon to enable the rootshell back door. Without further detail (such as a copy of inetd.conf), this is only an educated guess.

In case there was never any doubt about the presence of a rootkit, the kernel message "t0rns uses obsolete (PF_INET,SOCK_PACKET)" is a dead giveaway. The chkrootkit utility had already guessed that the t0rn rootkit might be installed, and here we see the word "t0rns" show up in the log. The message itself refers to the fact that whatever program was running in this case used an antiquated connection function, and the program is called "t0rns." T0rns is a packet sniffer that comes with the t0rn rootkit, which is logical, as the eth0 device enters promiscuous mode at the exact same time as the error message about the obsolete function. As is typical, the attacker is trying to sniff logins and passwords off the wire so he can attempt to root another box on the same network.

A few more lines into the log show a failed login attempt several hours later from a remote machine using "t0rn" as the username. Some time later there are some more inetd and syslog restarts, as well as an rlogin connection, which may or may not be coincidental. The IP was not in the same class as any of the earlier hosts' IPs, so it is difficult to determine whether these were related or merely coincidental. The failed logins could very well have been other attackers scanning for back doors created by their fellow attackers, or it could have been the same attacker trying to connect from another "owned" machine.

Analysis of the Incident

The staff responding to the incident made some fundamental judgment errors in their response that ultimately did not address the vulnerability and allowed the second attack to occur:

- The first time the student's machine appeared to be compromised, sysadmins should not have left the eradication step to an inexperienced Linux user with little knowledge of computer security. It was not surprising that the same compromise happened again only a few days later.
- The computing staff took appropriate measures and blocked the student machine's Internet access, but it should have remained blocked until they performed security tests against the machine, which would have revealed immediately that it was not yet secured.

- Determination of the attacker's IP address could have allowed the sysadmins to report the malicious activity to the service provider or address block the owner. This information can be obtained from IANA and/or WHOIS, and regardless of whether the IP belonged to another victim or to the attacker, it would have at least done a little to protect other Internet neighbors.

The event and the way university staff responded was probably as good as it could have been, given the following conditions:

- The computing services department is understaffed
- The computing staff, by necessity, devotes most of its time to the day-to-day user support required by administrators and faculty
- In a university environment, a large degree of network "openness" is critical
- Ease-of-use and flexibility (and the constant need to keep things working at all!) tend to end up prioritized over security

Ideally, if the resources existed, the support staff could take more extensive measures to prevent this sort of event, such as the following:

- Stricter tracking of which IP addresses are used by which machines;
- Grant the request for a static IP, and then, based on the information given by the student, apply appropriate filtering rules to the nearest router;
- Insist on hands-on configuration help for non-standard and/or multi-user OS's like WindowsNT/2000, Mac OS X Server, and UNIX/Linux;
- Slow down the log rotation (i.e. keep the logs for a longer period of time), and monitor the logs more closely and on a more regular basis;
- Filter more of the perimeter "noise" to help reduce the volume of data to be analyzed.

Finally, it the incident at XYZ University indicates a number of factors that would facilitate the spread of a SuperWorm. The detection and assessment of a vulnerability did not lead to a rapid response by either those being portscanned or by the university sysadmins. Additionally, the features of the network that allowed the compromise of the machine to occur have not been changed. The security measures of the university are neither sufficient to detect early indications of worm proliferation nor to stop the attack.

Section 3 - Early Detection of Active Internet Worms by Metering ICMP Destination Unreachable Messages

According to a Riptech study, cyber-attacks originate more often in the United States than in any other country, about thirty percent of all attacks.[23] The large number of Internet users in the United States may account for the high proportion; a recent estimate by New Scientist estimates that there are 166 million Internet users in the United States.[24] Nonetheless, analysis shows that lax security precautions taken by Internet Service Providers (ISPs) worldwide make it difficult to conclusively determine the exact source of the attack. Attackers can utilize the known vulnerabilities of a particular geography and launch or direct attacks through these networks.[25]

As indicated in the incident analysis of XYZ University, early warning of active worm propagation or other malicious activity over the Internet is of vital importance to first responders to cyberattacks. Identifying an active worm's characteristics very early in the propagation can reduce the damage it may cause. One security measure that could mitigate damage caused by malware is an early warning system that uses ICMP Destination Unreachable (ICMP-T3) messages to identify the random scanning behavior of worms. Participating routers across the Internet send Blind Carbon Copies of all their locally generated ICMP-T3 messages to a central collection point. Incoming messages are abstracted and patterns can be identified from the messages. Activity "blooms" will be identified that provide a clear signature of worm propagation.

Preliminary test results indicate that actively spreading worms can be identified in the first few minutes after they are launched. By examining the characteristics gathered in those early stages, actions can be taken and widespread damage might be avoided. When a worm scans for other vulnerable hosts, it hits addresses that are not actually represented by a physical host. This will trigger a router to return an ICMP-T3 Destination Unreachable message. When those routers send a copy of this message to a central point of collection, a clear pattern can be discerned.

It is possible to analyze ICMP destination unreachable messages generated by routers throughout the Internet to detect large-scale scans indicative of worm propagation, as well as other significant changes in pseudo-random target scan patterns. When a single host scans across the Internet for a certain vulnerability, a "bloom" of related activity is generated. When moments later one or more other hosts show the exact same kind of activity (in the form of blooms), scanning for the same vulnerability, then this is a clear sign of a worm propagating the Internet. By correlating detected scan events from a statistically significant number of participant routers worldwide, scan patterns will emerge indicative of high-priority events.

Data gathered during and after Nimda worm analysis shows that the vast majority of responses to connection attempts were ICMP Type 3 (ICMP T3) unreachable messages. Nimda used a random target selection algorithm weighted towards IP addresses numerically close to the source

[23] Kite, Shane, "Experts: Industry Must Confront Growing Cyberthreat," *Securities Industry News,* April 1, 2002.
[24] Knight, Will, "China's net usage leaps to second in world," *New Scientist,* April 23, 2002.
[25] Kite, Shane, "Experts: Industry Must Confront Growing Cyberthreat," *Securities Industry News,* April 1, 2002.

address. Ping and tcp port 80 scans used a non-weighted randomized scheme that avoided IANA reserved address space as determined via the unix command: whois iana@whois.arin.net.

The embedded content of the ICMP messages contains sufficient details of the failed datagram to ascertain the intended purpose of that datagram. A large number of these with similar attributes could be assigned to a unique scan event, or "bloom," and correlated against similar observations. This correlation process, taking place in near-realtime, will reveal new large-scale scan and worm activity.

This is an NIJ funded research project undertaken at the ISTS and is in the early stages of development, but researchers have achieved success in a test environment where the propagation of worms was simulated. As the research continues, a determination will be made about the best set of parameters and cooperation will be sought to gain more routers across the globe to get a more accurate view of real world worm propagation. For more information on the ICMP, please contact the IRIA for "Early Detection of Internet Worm Activity by Metering ICMP Destination Unreachable Messages" by George Bakos and Drs. Vincent H. Berk.

Section 4 - Modified Reverse Proxy Server

Nimda, Code Red, Code Red II, the IIS Unicode attack, and any other directory traversal attacks and/or input validation failures, could have been stopped through the use of the modified reverse proxy server (MRPS). While the damage caused by known incidents of malicious code is incalculable, and the threat of proliferation of a SuperWorm could cause wide-spread damage, the threat can be addressed and diminished on web servers through the implementation of a MRPS. This security technology will place a web server behind a network's firewall and deny access to the server. In doing so, those seeking to harm or alter information will be denied access to the server, as the MRPS will validate the request for information before making a request to the web server. In this manner, the MRPS adds another layer of security to the protection offered by the firewall. The firewall will authenticate the source of the request for information, and the MRPS will test and validate the request before interacting with the web server. Had this security technology been widely implemented before the recent attacks of worms and viruses, the damage could have been averted.

Networks are generally set-up as shown in Figure 5.

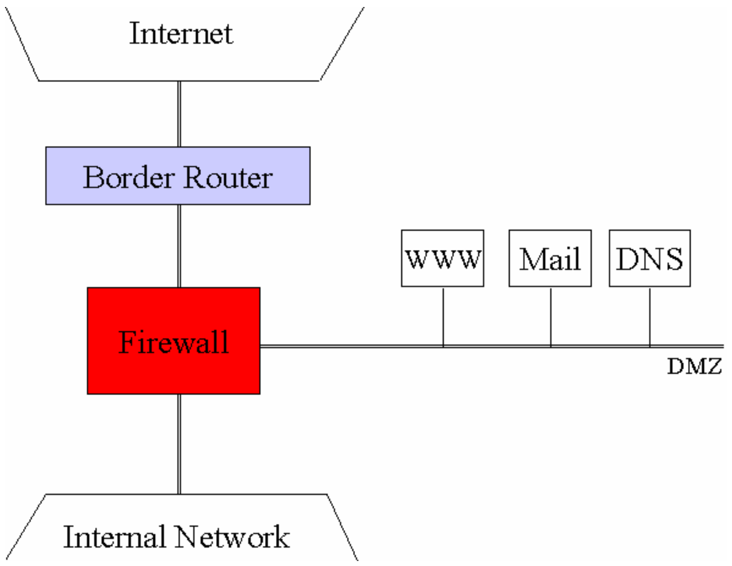

FIGURE 5

The connection with the Internet comes in through one or more routers, usually called *Border Routers*. Immediately behind the border router follows the firewall. Behind the firewall is the local network and on a third segment of the firewall are the public servers. The public servers usually are web (HTTP), mail (SMTP, IMAP, POP) and domain name resolution (DNS). If an FTP server is used, it is also commonly located in the public servers section of the network. This public servers section of the network is generally called the DeMilitarized Zone or DMZ.

The Modified Reverse Proxy (JEANNE), an NIJ funded research project undertaken at the ISTS, involves redesigning the network setup by putting the actual web server behind the firewall in the DMZ area of the server, and mirroring this server using reverse proxy servers in front of the firewall. The reverse proxy server will act as a web server; the reverse proxy server first handles all webrequests. Figure 6 provides an example of the revised network layout.

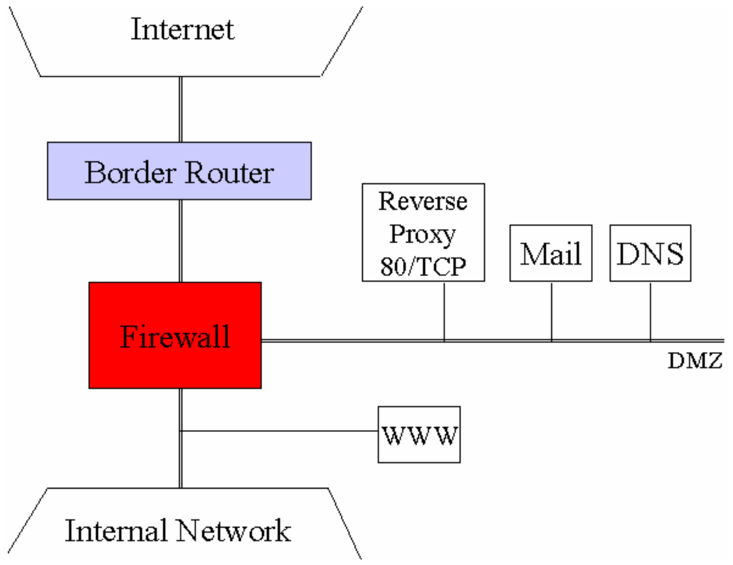

FIGURE 6

Due to the large number of different web servers available, all with a wide range of versions and security updates, it is very easy to become vulnerable to attacks. Also, due to increased complexity, configuration of modern day web servers can be confusing, leading to vulnerabilities due to misconfiguration. By placing the web server behind the firewall and denying access to this server, these weaknesses are no longer vulnerable to attack. The actual web server is mirrored using reverse proxy servers that are modified to do a wide range of extra checks on the incoming HTTP requests. If one of those tests fails, the request is denied. The reverse proxy server(s) get the WebPages through the firewall from the actual web server and cache them locally. This means that multiple MRPS can be used to mirror just one web server, which reduces the load and greatly improves the access time of the website. Subsequently, only one web server needs to be maintained while existing safely behind the firewall, decreasing vulnerability. Additionally, the MRPS can be utilized to mirror multiple web servers, enabling data from multiple machines to be accessed in a secure manner.

For a detailed description, please contact the IRIA for the white paper written by Vincent Berk, or see http://www.ists.dartmouth.edu/IRIA/projects/jeanne.htm.

The MRPS has been deployed not only at the ISTS, but also several clients. The research and testing are completed and the Beta version is stable. This concludes the research side of this project, however product dissemination is the next step.

There are products that are currently available that are similar to the Modified Reverse Proxy Server, JEANNE. Microsoft developed a product called "URLScan" (support.microsoft.com/default.aspx?scid=kb;EN_US;q307608).

The goal of URLScan is to scan an incoming request for several know attack patterns (for example, double dots, etc.) and validate the request in this way. The difference, however is that this product runs on the web server machine, which thus remains accessible from anywhere on the Internet. Subsequently it will only work on Microsoft's IIS server. JEANNE MRP will run on a separate machine, placing the actual web server securely behind the firewall. Also, since it runs on a separate machine, JEANNE MRP will work for any web server available.

The filtering mechanism of URLScan is based on known attacks and by default does not support Active Server Pages (ASPs) or other web server based executables (like CGIs). JEANNE MRP has support for both ASPs and CGIs. Whereas URLScan uses a default allow rule, JEANNE MRP will by default deny an access, this makes sense because if a request is unknown by the MRP, then the web server will not be able to handle it. This subsequently protects against still unknown vulnerabilities. In contrast; URLScan will let new attacks pass right through until its rulebase has been adjusted.

Squid is a web cache (proxy) server available under the General Public License (GPL).
They provide several links to security enhancing products, most notably DansGuardian and SquidGuard (www.squid-cache.org/related-software.html). Both security extensions work on a normal web-cache/proxy server. This is meant for outgoing connections and not for incoming. The Modified Reverse Proxy is specifically designed as a reverse proxy to validate incoming connections.

DansGuardian works by checking the URL for malicious phrases. This is to prevent users on your local network from contacting the outside world in a malicious matter. This could be for example a double-dot-slash if you do not want your local network to do this attack on the rest of the world. It is possible to prevent employees or children from accessing well know pornography sites or other sites with content deemed inappropriate. Even if this product were to be changed to work on a reverse proxy server it still does not cover new and unknown attacks. The JEANNE MRP will only allow requests that have been explicitly defined as valid. This is the main reason that the JEANNE Modified Reverse Proxy was the only tool that blocked the IIS Unicode Attack, Code Red, Code Red/II and Nimda by its very nature of design without any prior knowledge of either of these attacks and worms.

As with DansGuardian, SquidGuard only works on a standard outgoing proxy server and cannot be configured to be used as an inbound filter. Also it does not verify the actual request made and cannot be configured to do so. It merely operates on the user from the local network who is making an outbound request.

The importance of implementing the modified reverse proxy server is that it mitigates vulnerabilities caused by overloaded or less than diligent network security administrators. The security technology would not allow malicious traffic to pass to the real web server, and eliminate the threat posed by malware such as the SuperWorm

Conclusions

The problems faced by the systems administrators explored in the Incident at XYZ University are not unique, and the technologies described in subsequent sections are not the only means by which threats from the SuperWorm can be mitigated. For example, researchers of the IRIA are developing Tiny Honey Pot and the HoneyNet projects to gather data and detect anomalies. Both projects facilitate analysis of attack and pre-attack data to help researchers understand and formulate responses to threats, and the data is archived to assist law enforcement or security experts in the reconstruction of an incident for investigation. This data will help researchers discover the worm's origins, forensically reconstruct the path of the worm, and prepare a legal case against the worm author.

Unconventional and illegal means to address threats from worms have also been observed. For example, two different worms, Code Green and Crclean, were developed with the intention of patching Microsoft systems and removing the backdoor left by the Code Red worm.[26] Reactions by security experts were not positive, and stressed that regardless of intention, spreading a worm is still malware proliferation, and could have unintended consequences that cause additional problems.

Compounding the problem, there is no standard procedure for reporting vulnerabilities and addressing known exploits. When vulnerabilities are identified, the researcher or hacker decides whether to alert the software or hardware producer and give company representatives time to address the vulnerability, or publish their findings. Software vendors are recognizing the need for more reliable and secure products, but it remains to be seen whether adequate steps to create more secure products will be taken. Concurrently, creating patches has sometimes led to the creation of another vulnerability, and solutions can be as damaging as the problem. Finally, regardless of the speed by which the vulnerability is discovered and patches are released, the security of a system is ultimately dependant upon human behavior.

The cycle repeats itself, and original exploit can re-introduced over and over with counter-countermeasures to defeat the original steps taken to mitigate the threat. At what point, if any, does the threat become serious enough for the NIPC to recommend that machines should unplugged from the Internet?

Ron Dick indicated at the InfowarCon 2001 Conference on September 5, 2001:

> "President Clinton issued Presidential Decision Directive 63 in May of 1998. It was a wake up call which established a new framework for doing business. For the first time, the Federal government created an interagency entity, the National Infrastructure Protection Center-combining the United States law enforcement, military, and intelligence communities-to work directly with the private sector to achieve what many to this day say is impossible: The elimination of all vulnerabilities to our nation's critical infrastructures."[27]

It is in response to this mandate that the current cycle of vulnerability detection, assessment, and response is identified. If the theoretical SuperWorm was released into the current Internet environment, damage to critical infrastructure systems would be extensive.

[26] Middleton, James, " 'Anti-worms' Fight Off Code Red Threat," *VNUnet,* May 9, 2001.
[27] Dick, Ronald, "Legal Aspects of Infrastructure Protection," *InfowarCon 2001,* September 5, 2001.