A Worst-Case Worm *

Nicholas Weaver
International Computer Science Institute
nweaver@icsi.berkeley.edu

Vern Paxson
International Computer Science Institute
vern@icir.org

May 5, 2004

Abstract

Worms represent a substantial economic threat to the U.S. computing infrastructure. An important question is how much damage might be caused, as this figure can serve as a guide to evaluating how much to spend on defenses. We construct a parameterized worst-case analysis based on a simple damage model, combined with our understanding of what an attack could accomplish. Although our estimates are at best approximations, we speculate that a plausible worst-case worm could cause $50 billion or more in direct economic damage by attacking widely-used services in Microsoft Windows and carrying a highly destructive payload.

1 Introduction

Worms—malicious, self-propagating network programs—represent a substantial threat to the U.S. computing infrastructure, as they are capable of spreading substantially faster than humans can respond [22, 16, 14] and can contain highly malicious payloads [15, 26]. Since the development of automated systems designed to stop new worms represents a substantial investment in research and deployment, it is useful to estimate the amount of damage a worst-case worm could conceivably cause to assess how seriously society should take the threat.

We attempt to construct a defensible estimate of the possible worst-case damage the United States could suffer from a worm-based Internet attack. We use a methodology similar to a common approach used in risk management: we analyze the worst worm incident that is in our best judgment reasonably feasible for an attacker to implement.[1]

We combine our estimate of the worst-case worm with a simple linear damage model, based on lost productivity, repair time, lost data, and damage to systems. We exclude hard-to-estimate (and often grossly inflated) secondary losses and follow-on effects, and we also exclude possible impacts on critical infrastructure. We believe that our assumptions and model need to be transparent, allowing others to both evaluate our assumptions and develop their own damage models.

We assume an attacker with extensive available resources, such as a nation state. Our discussion is "defensible" in terms of attempting to anchor the analysis in concrete numbers concerning the possible spread of the worm and an explicit model of the damage it inflicts. The discussion is "worst-case" in that we assume that difficult-to-predict possibilities during the attack break favorably for the attacker. We do not make assumptions about which attackers might be motivated to actually carry out such an attack, just the capability required.

Although the ease with which computer attacks could disrupt power systems, dams, water treatment facilities,

*An earlier version of this work was performed under DARPA contract N66001-00-C-8045. This version was also supported by NSF grant ITR/ANI-0205519.

[1] Note that we have elided certain how-to details we worked out privately that might materially help an attacker while providing little additional insight for defenders.

and other such targets may be exaggerated [20], an electronic attack could cause widespread economic damage by disrupting or even destroying a large fraction of the computers responsible for day-to-day business. As we will develop, it is not implausible to conceive of attacks that could disrupt 50 million or more business computers, causing tens or perhaps hundreds of billions of dollars in direct damages. Thus, we assume an attacker with a specific goal to cause as much *economic* damage to the U.S. as possible by the immediate disruption of the U.S. commercial and governmental computing infrastructure.

The attacker might also wish to minimize the damage done in other countries, either to prevent the attacker's own economy from being disrupted (for technologically well-developed nation states) or to avoid galvanizing world opinion against the attacker. However, attackers without these constraints could just as readily target the entire world's Internet-connected infrastructure, with a side-effect being that the U.S. would be the most affected by such an attack. Outside of the U.S., it is mostly the other highly-developed economies which are vulnerable to this style of attack, as only those economies have a significant dependency on Internet-connected computers.

A weapon to achieve this is a sophisticated worm designed to infect as many machines as possible, spreading very quickly so as to infect all vulnerable machines well before human responses can intervene. The payload, when activated, would corrupt data on the infected machines and possibly damage hardware, as discussed below.

Additionally, based on previous experiences with the Slammer worm disrupting a nuclear power plant's systems [19], ATMs and 911 operations [5], and Welchia's disruption of the Navy Marine Corps Intranet [11] and ATMs [18], the attack could plausibly affect other critical infrastructure in hard-to-predict ways, which we can't account for in our model.

Unfortunately, our estimates on both penetration (the number of computers affected) and damage (the economic damage done on a per-computer basis) are necessarily crude. Our goal is not high precision. Rather, we desire to gain a feel for the magnitude of the incident, and whether the consequences are severe enough to justify substantial resources in prevention and mitigation. We believe that damage figures on the order of $50 billion or more are reasonably possible in a worst-case event.

2 Modeling a Worm's Damage

Our overall damage estimate is based on multiplying two factors, the number of systems penetrated and an estimate of the damage per system (Equation 1). This is a simple linear model: attacking N systems is N times worse than attacking a single system, which may miss important secondary effects, but the effects of these are hard to quantify and might go either way.

$$D_{total} = N_{inf} \cdot D_{system} \tag{1}$$

For evaluating the number of machines infected, we estimate a fraction of systems infected, multiplied by the total number of potential infectees (Equation 2). This depends on both the number of vulnerable machines and the penetration, or probability of infection. We estimate this figure in Section 3, where we evaluate the plausible worst-case attack and the necessary resources required.

$$N_{inf} = P_{penetration} \cdot N_{vulnerable} \tag{2}$$

Evaluating the damage done comes in four portions (Equation 3): the cost of system recovery, the productivity lost due to downtime, the value of any data loss times the probability of unrecoverable data loss, and the replacement value of the computer times the probability of system loss due to hardware damage (BIOS corruption). We estimate these parameters in Section 4, and evaluate how an attacker could increase the damage.

$$D_{system} = D_{rec} + T_{time} \cdot D_{time} + P_{data} \cdot D_{data} + P_{bios} \cdot D_{bios} \tag{3}$$

In section 6, we discuss the limitations of our model, before combining all the components in Section 5 to estimate the plausible damage from the worst-case attack. We also evaluate how sensitive our model is to these damage figures, by evaluating changes in our estimates.

3 The Attack

The goal of the attacker is to infect as many U.S. systems as possible, and to maximize the damage done to each infected system. The attacker's best tool is a worm, a self-propagating network program, designed to spread

as quickly as possible, maximize the damage done to the target systems, and remain active for as long as possible to reinfect any repaired but still vulnerable systems.

3.1 Attacker Resources

For purposes of this analysis, we assume the following resources: several experienced programmers highly knowledgeable of computer security, computer architecture, and existing systems; access to significant amounts of computing hardware for testing purposes; several months of time to develop and test the worm. For the specific attack we develop in this subsection, the task could go more quickly if the programmers have access to the source code for Microsoft Windows, but this is by no means a necessary precondition.

In our analysis, the main differences between an attacker with extensive resources, such as a nation state, and one with relatively limited resources, such as a terrorist group, is that the former can *(i)* attain more "zero day" (never-before-seen) exploits, and *(ii)* afford much more extensive testing. These translate into the extent of penetration into the vulnerable population that the worm will likely achieve, and the extent of the possible damage. We assume a nation-state class adversary in this discussion.

3.2 Candidate Service to Target

In order to attack as many targets as possible, the attacker must determine a widely-used service to exploit. An excellent candidate is *Windows SMB/CIFS file sharing*. This server is included as part of all Windows distributions since Windows 98, although the Windows 98 family (98/ME) and the NT family (NT/2K/XP) use different implementations. SMB/CIFS is used for desktop file sharing, printer sharing, and by centralized Windows file servers. In addition, the service is on by default on most installs. For this analysis, we assume the attacker knows a SMB/CIFS exploit which enables arbitrary remote execution.

Several factors make SMB/CIFS particularly attractive:

- widely deployed
- affects both servers and workstations, so the attacker can potentially target the entire corporate computing infrastructure in a single attack
- includes default anonymous login capabilities, which enables some exploits to connect without requiring authentication
- workstation users can authenticate to other workstations and servers within a domain
- vulnerabilities have been discovered in the past
- since the SMB service runs as part of the OS kernel, any successful attack gains complete control of the machine
- the on-by-default nature of the service implies than if an exploit is discovered, most Windows PCs are automatically vulnerable
- since file sharing is often critical to business operations, organizations cannot lightly disable it.

For the attacker, the greatest concern is the size and complexity of the protocol.

A major vulnerability in SMB/CIFS was discovered in the summer of 2003 [12]. This vulnerability could allow arbitrary remote execution as long as the attacker is authenticated within the domain. Likewise, the RPC vulnerability [13] used in the Blaster worm [23] represents a similar exposure, and could have also been used as the basis for this analysis. (Indeed, we first drafted this paper—including targeting SMB/CIFS—before either of these vulnerabilities had come to light.)

Discovering an otherwise unknown vulnerability ("zero day" exploit) in this service will create the foundation for a devastating worm. Since the exploit is otherwise unknown, effectively all Windows systems would be vulnerable, regardless of whether they are properly patched and maintained.

When the worm compromises a machine, it queries the local Windows domain controller to receive a list of all the local machines and their names. Using this information, the worm can quickly compromise all machines in its domain within seconds (i.e., a "metaserver" worm [25]). To compromise more machines, the worm then scans for vulnerable targets in the corporate intranet before beginning to scan the general Internet.

These services are particularly vulnerable even to old attacks. The Blaster worm was released almost a month

after the highly publicized RPC vulnerability was discovered. Yet, despite massive publicity beforehand, Blaster infected, at minimum, 8 million systems [9].[2]

3.3 Crossing Firewalls and Penetrating Domains

Since the SMB/CIFS service (or the related RPC service) is usually used only within the corporate intranet, most institutional firewalls will prevent incoming connections. Likewise, for most SMB/CIFS vulnerabilities the attacker must be authenticated within the domain, which poses similar barriers.

Thus, the attacker will probably require a different mechanism to obtain the initial penetration of the corporate intranet and Windows domain. To do so, they can adapt the approach used very successfully by Nimda [4, 22] of first spreading rapidly across the Internet using a scanning technique, and then including secondary modes such as a mail-worm mode or an infected web-server mode that can infect browsers inside internal networks. Another option is to use known vulnerabilities in otherwise externally-accessible services (such as the IIS web server) as a means for the attacking worm to transition into the domain. A final option is to take advantage of notebooks and wireless access cards.

This strategy is effective for several reasons. In most cases, a worm attacking SMB/CIFS only needs a single compromise to spread through the entire corporate intranet or large Windows domain, suggesting that a relatively low volume of mail needs to be sent. Although mail worms and related attacks only run with the privileges of the victim, the primary SMB/CIFS exploit could be used to gain full system privileges by connecting through the SMB service on the local host using the local user's authentication. Furthermore, while mail-worms can benefit from vulnerabilities in mail-readers, they do not require such vulnerabilities if a user can be enticed to run an executable attachment. Finally, the current mail worm defenses, both on servers and workstations, are based on signatures. Thus, new mail worms slip through most mail-servers, client anti-virus programs, and firewalls during the first few hours of spread.

Another strategy, also employed by Nimda, is to exploit known flaws in common web browsers to compromise the browser's host. When the worm infects a machine that can write to `.html` files, it changes the files so that when a vulnerable web client attempts to access information, the client will download and execute the worm. Again, this mode does not need to be widely successful, as a single penetration would usually suffice to enable the SMB mode to corrupt the corporate intranet.

Finally, an attacking worm could recognize that it is running on a notebook, and use these as additional resources. Rather than being heavily aggressive, a notebook infection could be mostly silent, only contacting the local neighborhood when connected to a network. Since notebooks frequently cross trust-boundaries, this would enable the worm to penetrate firewalls. Additionally, any system with a wireless card could be used to search for wireless networks [27].

3.4 Targeting the Worm

If the attacker desires to target the worm against U.S. interests, there are several tactics that could be employed to minimize collateral damage. When scanning across the Internet, the worm could use well-known information about the structure of IP address allocations to restrict its scanning to U.S. addresses [28]. A mail worm could restrict itself to U.S.-related top level domains, or use the current selection of language or localization features to control collateral damage. Finally, the worm could instead spread without restriction but then confine its later destructive behavior to machines whose localization settings or IP address suggest a U.S. origin.

3.5 Speed of Propagation

We analyze the three phases the worm would spread in: spread across the open Internet, spread through firewalls and other gateways into intranets, and then spread across intranets. These phases would overlap to some degree.

3.5.1 Internet Spread

The Slammer worm [14] demonstrated the ability of a worm to spread across the Internet and infect 10's of thou-

[2] This figure probably represents an undercount, because it only includes infected systems that contacted Windows Update and downloaded a removal tool, not any infected systems that were simply reinstalled, patched, and then reconnected to the Internet.

sands of servers in less than ten minutes. On theoretical grounds, we had earlier predicted Internet worms that could spread in less than a minute ("flash" worms) [22].

Thus it should be assumed that a worst-case worm could acquire a very sizeable population of Internet servers in minutes at most. This population could then be used to launch the next phase of the spread. The speed of this first phase means that organizational gateway defenses must be assumed to be unprepared at the outset (for example, no signatures available for mail gateway antivirus checks).

3.5.2 Spread through gateways

When spreading from the Internet to within corporate intranets, the mail and web vectors are slower since they depend to some degree on human actions within the organizations. This phase is likely to dominate the overall spread time of the worst-case worm.

It is difficult to estimate this spread time with precision since data for worm spread within organizations is not generally available. The best estimate available to date is from the Nimda worm's firewall penetration routines, where the evidence suggests that the worm spread within a few hours [22], despite the fact that its scanning mode was much slower than that of Slammer.

A very conservative upper bound would be based on recent pure mail-worms such as SoBig.E [10], which required a little more than a day to reach peak volume. The actual reality will probably be much faster, as only a single penetration is required to infect an institution.

3.5.3 Intranet Spread

Within a corporate intranet, complete infection could well be nearly instantaneous. With 100 Mbps and 1 Gbps LANs, infecting a new victim will require less than a second. Due to the exponential growth of such a worm, complete infection of the intranet would happen in much less than a minute. This is made worse by the large number of vulnerable hosts that, paradoxically, enables worms to spread faster [22].

This is especially true with a CIFS vulnerability that exploits the metaserver properties of the domain controller (i.e., asks the domain controller for the addresses of other likely-infectible hosts). In this case, no scanning is required at all, instead the worm simply attacks all the victims in the institution in a matter of seconds.

3.5.4 Total Spread Time

Although it is probably impossible to estimate more precisely, our experience suggests that such a worm could propagate to all areas of the organizational networks where users are active within a few hours, with the time dominated by the requirement to cross the organizational gateway. Thus, if released during U.S. business hours, it could infect all the vulnerable machines before a reaction is possible, as even the highly disruptive and detectable Slammer worm [14] was effectively unperturbed for 3 hours.

To ensure such rapid spread, there are some tricks an attacker could employ. One effective technique (providing the attacker takes care to avoid leaving tracks) involves mailing (spamming) several hundred or thousand copies of the mail worm, targeting numerous large corporations to create a "hit list"-style effect [22]. There is some circumstantial evidence suggesting that mail-worm authors are already using this technique [6].

3.6 Testing

The biggest hurdle an attacker faces is not creating the worm but testing its functionality. The network code and propagation routines are reasonably generic, using well documented APIs, and should therefore work on most or all target systems. However, the malicious payloads and exploit code tend to be much more specific: minor changes in the system may render the code unusable.

Thus, considerable attacker effort needs to be spent in testing these components in a wide range of environments. The more diverse the testing, the more widely the resulting worm is likely to penetrate.

There is a precedent to suggest that a determined and well-funded attacker could make such code work on a wide variety of systems. A non-malicious example from a similar domain is the Macrovision SafeDisc 2 system, which embeds a digital signature in a non-standard way on the CD, preventing typical CD-burners from duplicating the disk [7]. The signature is then used to verify that the proper disk is in the drive when the program runs.

Although derided for what are in fact infrequent incompatibilities, it is able to run on most systems even when accessing the CD drive using nonstandard techniques to read deliberately "erroneous" data.

Additionally, there would be great value to the attacker in testing the ability of the worm to elude the market-leading anti-virus and intrusion detection systems on release. This would ensure that no effective countermeasures would prevent its spread until after it had been analyzed and signatures deployed. To the extent the attackers could make the worm polymorphic, or include specific anti-anti-virus routines, this would further undermine defense mechanisms.

3.7 Estimating The Number of Infected Systems

Given such a highly virulent worm, we now turn to estimating the total number of machines which might be compromised. A worm would be unable to attack all of these, because:

- some networks and computers are not connected to the Internet,
- other networks and computers might have sufficient defenses, especially if their gateways can be supplied with signatures that prevent the worm from entering,
- the worm's SMB exploit may not work on all possible combinations of hardware and Windows.

The degree to which the attacker can control these last two variables is largely a function of how thoroughly the worm can be tested.

For a nation state adversary, with access to a very extensive test environment and with particularly thorough testing, we estimate penetration of approximately 60% of the vulnerable business PCs is a plausible worst case ($P_{penetration} = 0.60$). This is based on the attacker using zero-day exploits, testing his attack well, and our observations on how worms interact with conventional defenses [27].

Another observation regarding penetration, not considered in our further analysis, is that even with relatively low penetration, the described worm would have a disproportionate effect on large institutions, as there are more opportunities to gain the initial penetration on larger networks with more users.

We also need to estimate the number of possible systems affected. One survey conducted in 2001 suggests there were over 85 million PCs in U.S. businesses and governmental use [1], with an equivalent number in U.S. homes. Thus, with $N_{vulnerable} = 85{,}000{,}000$, Equation 2 gives us $N_{inf} = 50{,}000{,}000$ affected systems.

We are also neglecting the effects on home machines. The U.S. Census Bureau [3] estimates that there were 44 million households in the U.S. with Internet-access (at least one personal computer and either a dialup or dedicated Internet connection) in 2000, but for purposes of this analysis, these systems are considered to have zero value.

4 The Attack's Damage

The damage done arises from four sources: the cost of restoring the system to normal operation, the cost of lost downtime, the value of any data lost, and the replacement cost of the system if the system is irrevocably damaged through hardware damage (BIOS corruption). The attacker desires to increase all these costs by using a highly malicious payload.

4.1 Data Damage Payload

As soon as a worm compromises a machine, it can begin acting in a malicious manner as long as such behavior does not slow the spread of the worm, and, ideally, does not invite detection. The worm can first install a Trojan in the boot block for later activation. It can then search remote and local drives, randomly corrupting files [15, 26].

After the worm has decided that the infected machine is no longer needed as part of the spreading process, either by using a post infection timer, determining that it can't infect any more machines, or other techniques [22], the worm can activate malicious behavior that is overtly detectable and would otherwise inhibit its spread.

One type of damage easy to effect is to comprehensively erase all remote and local disks.[3] An effective strategy would be to initially overwrite random sectors on the

[3] Written before the release of Witty, which inflicts damage similar to this [15].

disk to ensure a general corruption, followed by a comprehensive erase routine which requires considerably more time.

4.2 Hardware Damage

The damage payload might also attempt to reflash the BIOS,[4] corrupting the bootstrap program used to initialize the computer.

We conducted a small survey of manuals and other web-based sources for 7 popular systems and 2 motherboards (specifics elided). The documentation for all of these systems suggests that the BIOS is flashable by software in the default configuration.

The actual routines required are vendor-specific, forcing the attacker to decide on a subset of machines to target. However, since the programs involved are readily available and can update many configurations, an attacker could develop a tool to perform automated analysis of BIOS updates in order to create a library of routines needed to reflash many different motherboards.

In one third of the cases we examined, it appears possible for the worm to cause damage to the hardware that would require motherboard replacement before the system could function again. In the other two thirds of the cases, it is possible to corrupt the BIOS but the BIOS can be restored via a complex procedure that usually requires opening the computer, and is beyond the skills of most computer users and perhaps even many system administrators.

4.3 Attempting Reinfections and Increasing Downtime

The use of a zero-day exploit significantly increases the downtime, especially if the systems are vulnerable in the default configuration. This is especially catastrophic if the exploit is in Windows file-sharing and the worm attacks the local neighborhood: it would require significant time for either patches or workarounds to be deployed.

An additional problem which sysadmins face is reinfection. The time between when a system is restored and when a patch is installed allows a system to be reinfected if there are still copies active on the local network. Often, even just a minute of vulnerability is enough to allow reinfection. Thus, reinstallation may need to occur offline, with patches manually loaded onto systems.

A worm that is highly malicious to the host will eventually become extinct [15], which limits opportunities for reinfection. One workaround is for a small percentage (say 5%) of the infected systems to behave in a more benign manner, in order to keep an active reserve of infection.

4.4 Estimating Damage

The first term in Equation 3, D_{rec}, represents the system administration time to restore the system: reload the operating system, install patches, reinstall applications, restore data from backups, and reconnect the system to the network. Some institutions use remote install techniques or similar services but, even then, significant administrator intervention may be required. Where remote-install is not already preconfigured, installation of a single machine may take several hours of administrator time.

For purposes of this analysis, we will assume that it only requires 1/2 hour of system administrator time to restore the system, as most institutions will use mass-install techniques, or have administrators parallelize a manual install by fixing several computers simultaneously. Assuming a $50,000 annual salary, with an additional 50% cost in payroll taxes and benefits and a 50 week work-year, an hour's time for a system administrator is roughly $40. Thus we set $D_{rec} = \$20$ per system.

The second term, productivity loss due to downtime, depends on both the value of the labor and the time lost. For D_{time}, we took the entire value of the U.S. GDP ($11 trillion), divided by the number of workers (138 million) in the U.S., who work an average of 34 hours per week [2], giving us a value of approximately $45/hr as the average productivity per U.S. worker.

Yet not all this time would really be lost, as there are undoubtedly other, noncomputer related tasks which could be performed. Thus we set $D_{time} = 35$ $/hr to compensate for this observation. We then assume that $T_{time} =$ 16 hr, that is, average downtime is two days per user for our base assumptions. This represents a single day for Microsoft and others to reverse-engineer the worm and to develop patches and workarounds, and a second day for the local system administrators to restore full network op-

[4] A technique first seen in the Chernobyl [8] virus.

eration. Lost time is, by far, the most significant source of damage in our model.

The third term, lost data, is harder to quantify. In [21], the estimated cost of a single data loss incident is $2,500, with ≈ $2,000 representing the average value of the data itself[5] (assuming it is eventually recovered 80% of the time) and about $500 in lost productivity and technical service time to restore the computer. Although an arguable figure, we set $D_{data} = \$2{,}000$.

For purposes of our analysis, we assume that the data is *not* lost most of the time, despite the attacker's intent, because there are suitable backups. We also accounted for lost-time and system administrator cost in other portions of our model. Thus, we set $P_{lost\_data} = 0.1$. This is conservatively lower than that used in [21] as several infections might constitute only a single data loss event.

Our final term, the cost of lost systems due to corrupted BIOSes, depends on the attacker's skill and testing. Even for a nation-state adversary, we assume that $P_{bios} = 0.1$, as the attacker can only permanently destroy a limited number of configurations. Undoubtedly the attacker will develop automated tools to increase the number of systems affected, but it seems unlikely that he can attack significantly more than 10-20% of the vulnerable population.

However D_{bios}, the cost of each system permanently damaged, is high. Not only is there the cost of a replacement system (which we assume as $1,000), but there is a substantial increase in lost time, as although the computer business is robust, manufacturing lead-times are relatively long and inventories are low. Thus for all machines out of commission, we add an additional $1,400 to account for an additional 40 hours of lost productivity, giving us $D_{bios} = \$2{,}400$.

5 Estimating the Loss

Table 5 has our combined estimates for various scenarios. The first is our set of baseline assumptions (2 days lost productivity, 10% permanent data damage, 10% permanent BIOS damage). We believe this represents a conservative figure, as we only attempt to estimate direct costs, using what we believe are conservative values for the parameters in our model.

[5] [21] states that this figure is particularly hard to estimate.

The second assumes that, rather than 2 days downtime, the attack causes 4 days downtime for most users. We have directly observed the problems of reinstalling systems in a hostile environment, even when patches are available. Even supposedly "patched" install images have sometimes proven vulnerable in the past, as the patches are only activated late in the installation process.

The third model, increased data damage, assumes that data is lost 20% of the time rather than 10%. The fourth model postulates increased effectiveness for a BIOS/hardware damaging attack, with 20% of the infected systems disrupted. Although both forms of damage are significant, overall they have less of an impact than lost time. The final model demonstrates what happens if all the significant damage components are doubled: 32 hours of lost time, 20% data loss, and 20% hardware damage.

6 Other Effects and Model Limitations

Our model is limited in several ways: it does not consider the nonlinear effects on lost time, the possible severe impact on some companies, follow-on effects and secondary damage, or possible damage to critical infrastructure.

Not all lost-time is equal. A downtime of an hour is of little consequence for most workers as there are always noncomputer tasks. A downtime of a day is more significant, while several days might be hugely disruptive. Yet we have no way of effectively estimating this phenomenon, so we exclude it from our model.

Likewise, most defenses against worms of this class are brittle: either they succeed, and an institution is unaffected, or they fail, and the entire institution is disrupted. The same phenomenon will likely apply with a data-damaging or BIOS-reflashing payload, due to institution-wide policies and purchasing decisions. It is an open question whether this phenomenon might amplify the damage, as some companies may be entirely unaffected, while others might suffer weeks of lost productivity.

Our model does not consider follow-on consequences, just the direct impact of the lost time. We would be ill-advised to consider these effects because we do not know how to accurately model them. We are also reluctant to

Attack & Assumptions	N_{inf}	D_{rec}	T_{time}	D_{time}	P_{data}	D_{data}	P_{bios}	D_{bios}	D_{system}	D_{total}
Baseline	50 M	$20	16	$35	0.1	$2,000	0.1	$2,600	$1,040	$52 B
Increased Downtime	50 M	$20	32	$35	0.1	$2,000	0.1	$2,600	$1,600	$80 B
Increased Data Damage	50 M	$20	16	$35	0.2	$2,000	0.1	$2,600	$1,240	$62 B
Increased BIOS Damage	50 M	$20	16	$35	0.1	$2,000	0.2	$2,600	$1,300	$65 B
Increased All	50 M	$20	32	$35	0.2	$2,000	0.2	$2,600	$2,060	$103 B

Table 1: Estimated Damage for various attack scenarios.

consider them because of the tendency for these values to be inflated.

The biggest limitation is our inability to evaluate possible damage to critical infrastructure, as there is a non-negligible chance that such a worm might also affect the power grid, hospital systems, telecommunications, or other systems. Slammer managed to infect an off-line nuclear power plant control computer [19], disrupt Bellevue Washington's 911 system, and Bank of America teller machines [5]. Likewise, Welchia managed to directly infect Diebold ATMs [18], and disrupt the Navy-Marine Corps Intranet [11] while the United States was engaged in substantial military action.

Undoubtedly, a worm similar to what we describe might infect these systems as well, depending on the worm's ability to penetrate the more substantial defenses protecting such networks. Assuming a zero-day vulnerability, such penetration works could cause substantial disruption to any networked, Windows-based critical systems.

7 Current Defenses and Recommendations

Current defenses are not capable of dealing with threats of this magnitude. Most email worms are stopped through signature-based scanning, a technique that is both easily avoidable and cannot detect new worms. Similarly, most intrusion detection systems are focused on signature-based methods and are only deployed to protect a corporate intranet from external attack, while the proposed worst-case worm infects most machines through internal connections. None of these defenses are capable of dealing with threats that can propagate nationwide in an hour or less.

Mail worm vectors can largely be eliminated by a matter of policy, if restrictive policies can be employed. In addition to employing virus scanning on incoming email, all executables could be quarantined for several hours, enabling the scanners to be updated if new threats arrive. This quarantine period should be long enough that signatures will be updated in the presence of a new worm.

Additional filters could search for unusual characteristics, such as excessively long strings in headers, which may be required by exploits targeting mail readers. Such messages could either be quarantined or modified before forwarding.

A substantially harder task is preventing an SMB/CIFS, RPC, or similar worm from spreading throughout the corporate intranet. The best defense requires restricting the network topology either at the switches or using desktop firewalls. For some users, these restrictions may be unpalatable, but they offer a substantial defense.

If instead of enabling any machine to access any other machine's file sharing service, the network could be restricted so that file sharing and related services on user desktops can only be accessed from dedicated administration machines running no services at all. This prevents an infected desktop or server from compromising other

desktops, greatly limiting the spread of the worm, as now only servers can be infected from a compromised system. An alternate approach that may prove effective is to restrict the Windows "active directory" server to limit authentication so that workstations cannot respond to most connections.

Even with such restrictions, it may be possible for an infected server to compromise a client. This would most likely probably require an additional exploit, as the asymmetric nature of communication prevents most contagion [22] strategies from operating.

If the servers themselves are not Windows systems, instead using SMB/CIFS-compatible servers such as Samba [24] under Linux or a Network Appliance file server [17], this effectively halts desktop-to-server and server-to-server transmission, as infection requires different vulnerabilities unless the author constructs a multi-platform worm.

Combined with the topological restrictions, this should isolate most possible SMB/CIFS worms, preventing them from spreading through the intranet, as the client-to-client, client-to-server, server-to-client, and server-to-server transmission paths should be broken by these changes. Yet this requires substantial network rearchitecting, which may not be feasible in current environments.

Other techniques can help limit the total damage. Most motherboards include jumper or switch settings that disable BIOS reflashing. The disadvantage of write-protected BIOSes is that it requires opening the case (and resetting the jumper) to reflash the BIOS, a possible inconvenience. Setting the jumper in the first place also requires opening the case, as most systems do not ship with write-protection enabled.

Data can be protected by standard storage management policies: nightly backups, file servers that support snapshots, and off-site storage protection. Thus, unless the worm also includes mechanisms to erase the backup media, most data will not be permanently corrupted. For both convenience and ease of administration, it is probably best to use centralized fileservers and standard (quickly reformattable) clients that don't maintain critical local state, in an effort to speed recovery.

Beyond prevention, it is critical to develop and deploy automated systems designed to detect, analyze, and counter novel worms before most systems are infected. This is an active area of research and development.

8 Conclusions

Such an attack, plausibly representing fifty billion dollars or more in direct damage—and with difficult-to-estimate but quite possibly large additional indirect damages—would cause serious harm to the U.S. economy. A non-violent attack that can cause such economic distress is of significant interest to numerous parties, including nation states or groups engaged in economic competition with the United States, as well as terrorist organizations.

There are preventative measures: protecting BIOSes, solid backups, mail-worm defenses, modifying topologies, improved recovery procedures, and reducing monocultures, which significantly mitigate the potential damage from this sort of worst-case worm. Finally, although there are other vulnerable "ecologies" that can support very widespread worms, SMB/CIFS (and the related Windows RPC service) is particularly ubiquitous and therefore a highly attractive target meriting somewhat specialized defenses.

9 Acknowledgments

Stuart Staniford is a coauthor of this work. When revising the paper for final copy, we found a deep disagreement regarding the presentation and interpretation of the possible damage (Stuart believes a sound case can be made for it being substantially larger, and that it is irresponsible to not include this in the revision), which we were unable to resolve by the camera-ready deadline. We much regret that the only way we were able to still have the paper appear in the workshop was by withdrawing his name from the authorship of the paper. We respect the heartfelt principles that led him to his position.

Our thanks to Rob Cunningham, Stuart Schechter, and the anonymous reviewers for many helpful comments and ideas.

References

[1] Computer Industry Almanac. PCs in use surpass 600M, http://www.c-i-a.com/pr0302.htm.

[2] Bureau of Labor Statistics, The Employment Situation: http://www.bls.gov/news.release/pdf/empsit.pdf.

[3] Home Computers and Internet Use in the United States: August 2000, http://www.census.gov/prod/2001pubs/p23-207.pdf.

[4] CERT. CERT Advisory CA-2001-26 Nimda Worm, http://www.cert.org/advisories/ca-2001-26.html.

[5] Richard Forno. Lessons From the Slammer, http://www.securityfocus.com/columnists/140.

[6] Gregg Keizer. Virus-Writers Using Spammer Techniques to Speed Spread, http://www.internetweek.com/security02/showArticle.jhtml?articleID=10300197.

[7] Macrovision Inc. SafeDisk Whitepaper, http://www.macrovision.com/solutions/software/cdrom/safediscwp4-17-02.pdf.

[8] Kaspersky Labs. W95/CIH (a.k.a Chernobyl), http://www.viruslist.com/eng/viruslist.html?id=3204.

[9] Robert Lemos. Msblast epidemic far larger than believed, http://news.com.com/2100-7349_3-5184439.html.

[10] Message Labs. Message Labs Threats Information, http://www.messagelabs.com/viruseye/info/default.asp?virusname=W32%2FSobig.E-mm.

[11] Ellen Messmer. Navy Marine Corps Intranet hit by Welchia Worm, http://www.nwfusion.com/news/2003/0819navy.html.

[12] Microsoft. Microsoft Security Bulletin MS03-024: Buffer Overrun in Windows Could Lead to Data Corruption, http://www.microsoft.com/technet/security/bulletin/ms03-024.asp.

[13] Microsoft. Microsoft Security Bulletin MS03-026: Buffer Overrun in RPC Interface Could Allow Code Execution, http://www.microsoft.com/technet/security/bulletin/ms03-024.asp.

[14] David Moore, Vern Paxson, Stefan Savage, Colleen Shannon, Stuart Staniford, and Nicholas Weaver. Inside the Slammer Worm. *IEEE Magazine of Security and Privacy*, pages 33–39, July/August 2003 2003.

[15] David Moore and Colleen Shannon. The Spread of the Witty Worm, http://www.caida.org/analysis/security/witty/.

[16] David Moore, Colleen Shannon, Geoffrey M. Voelker, and Stefan Savage. Internet Quarantine: Requirements for Containing Self-Propagating Code. In *INFOCOM*, 2003.

[17] Network Appliance. NetApp Filers, http://www.netapp.com/products/.

[18] Kevin Poulsen. Nachi worm infected Diebold ATMs, http://www.securityfocus.com/news/7517.

[19] Kevin Poulsen. Slammer Worm Crashed Ohio Nuke Plant Network, http://www.securityfocus.com/news/6767.

[20] Robert Lemos. What are the real risks of cyberterrorism? http://zdnet.com.com/2100-1105-955293.html.

[21] David M Smith. The Cost of Lost Data, http://www.legato.com/resources/whitepapers/W052.pdf.

[22] Stuart Staniford and Vern Paxson and Nicholas Weaver. How to 0wn the Internet in Your Spare Time. In *Proceedings of the 11th USENIX Security Symposium*. USENIX, August 2002.

[23] Symantec. W32.blaster.worm, http://securityresponse.symantec.com/avcenter/venc/data/w32.blaster.worm.html.

[24] The SAMBA Project. Samba: Opening windows to a wider world, http://www.samba.org/.

[25] N. Weaver, V. Paxson, S. Staniford, and R. Cunningham. A Taxonomy of Computer Worms. In *The First ACM Workshop on Rapid Malcode (WORM)*, 2003.

[26] Nicholas Weaver and Dan Ellis. Reflections on witty: Analyzing the attacker. *;login:, scheduled to appear*, 2004.

[27] Nicholas Weaver, Dan Ellis, Stuart Staniford, and Vern Paxson. Worms verses perimiters: The case for hard lans, in submission.

[28] Cliff Zou, Down Towsley, Weibo Gong, and Songlin Cai. Routing worm: A fast, selective attack worm based on ip address information. umass ece technical report tr-03-cse-06.