available at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Functional similarities between computer worms and biological pathogens☆

*Jun Li**, *Paul Knickerbocker*

Department of Computer and Information Science, University of Oregon, 1477 E. 13th Ave., Eugene, OR 97403-1202, USA

ARTICLE INFO

Article history:
Received 5 September 2006
Revised 7 December 2006
Accepted 11 December 2006

Keywords:
Computer worm
Worm detection
Worm defense
Biological pathogen
Biological infection
Biological similarity
Operational exploit
Environmental exploit

ABSTRACT

Computer worms pose a serious threat to computer and network security. Interestingly, they share many common tactics with biological pathogens with respect to infecting and propagating. In this paper, we study the six most common fatal infectious diseases—measles, malaria, HIV/AIDS, tuberculosis, influenza and the diarrhoeal diseases—to (1) determine the individual mechanisms and environmental conditions that have contributed to their success, and (2) show the parallels between the mechanisms and behavior of successful biological infections and successful digital infections. Moreover, by identifying the specific areas of similarity and looking at effective preventive and creative measures used against biological pathogens, we draw insights about what steps individual computers and networks can take to protect themselves.

1. Introduction

Fighting computer worms is a critical but daunting task. The potential for damage from computer worms has increased in direct relationship to the importance of legitimate software in our lives. The stakes in the fight between security professionals and malicious worm programmers have been rising steadily, as has the ingenuity of these programmers. In the biological world there has been a similar ongoing struggle between organisms and the infectious agents that prey upon them. It is an arms race that has been under way for millions of years with the ultimate stakes: life or death. Using the knowledge that we have gained through countless studies of the diseases that attack humanity, we can better understand the computer worms and viruses that attack our computers. This will enable us to discern the conditions that lead to vulnerability to such attacks and to come up with new and superior countermeasures.

Whereas there have been a plethora of studies based on a biology–computer analogy for defense methodologies (Section 2), there have been comparatively few studies on the intrinsic similarities of the attacks themselves to biological systems. Biological terms such as "worm," "virus," or "rabbit" have been borrowed to name computer attacks, but the research in this area has mostly focused on leveraging epidemiological studies of disease propagation to predict computer worm and virus propagation (Kephart and White, 1991, 1993; Murray, 1998; Williamson and Léveillé, 2003). Studies on the

☆ This research is partially supported by Intel Corporation.
* *Corresponding author*. Tel.: +1 541 346 4424; fax: +1 541 346 5373.
E-mail addresses: lijun@cs.uoregon.edu (J. Li), pknicker@cs.uoregon.edu (P. Knickerbocker).
0167-4048/$ – see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cose.2006.12.002

intrinsic characteristics of biological pathogens and how those relate to computer worms are mostly informal and ad hoc. This leaves a significant gap in our ability to leverage the knowledge and experience of the biology community in defending ourselves against computer worms and viruses. The research presented in this paper addresses this gap by enumerating the functional similarities between biological pathogens and their computer counterparts, allowing us to gain insight into desirable characteristics and potential methods of effective defense.

The main question we are concerned with is: *What can the behavior of successful biological infections teach us about infections in the digital world?* While we recognize the inherent differences between biology and computers—connecting to a communications port has almost nothing in common with a biological virus connecting to a protein receptor on a cell membrane—infection vectors in both the biological and computer worlds are little more than self-replicating pieces of code. Although one uses genetic material and the other uses a series of computer instructions, the two follow similar patterns in the way in which infections are transmitted and in their behavior inside infected hosts.

Specifically, whether the infection is biological or digital, both focus on subverting complex systems through weaknesses in design or environment. In other words, systems can be subverted either through weaknesses in their own implementation, or weaknesses inherent in the environment in which they operate. *Operational exploits* target weaknesses that are inherent in the construction and operation of systems, and *environmental exploits* take advantage of the weaknesses created by adverse conditions and failures in defenses of the systems.

Our study provides a new framework in looking at computer worm attacks. By studying the operational and environmental exploits that biological pathogens have been employing for ages, we will show how computer worms—which only began to appear less than two decades ago—may cause severe devastation by using essentially similar infection techniques and factors. Our study could also help identify certain worm techniques that have drawn little attention. In particular, this study demonstrates that similar to biological pathogens, worms can piggyback on legitimate entities, explore topology information, incubate, become polymorphic, and target critical resources or defenses, while benefiting from an infection-friendly environment with digital pollution, monoculture, unpatched systems, and user complacency. Finally, our study is primary to leveraging biological means for computer worm defense. While many computer defense approaches already borrow ideas from biological world, only after a solid understanding on functional similarities between computer worms and biological pathogens can countermeasures against biological pathogens be effectively leveraged in defending against computer worms.

The rest of this paper is organized as follows. We first present the studies related to ours in Section 2. Then in Section 3 we discuss the scope of our approach, narrowing down what attributes of biological infections we will review and how we will present the analysis. Section 4 explores the individual techniques of specific infections that have proven successful across different environments and in the face of deliberate actions taken against them. This is followed by Section 5 in which we describe biological techniques that have proven successful because they exploit flaws in the environment in which the infection operates. We conclude the paper in Section 6 with an interpretation of six main insights and their implications for security practitioners.

2. Related studies

The similarity between biological processes and computer security problems has long been recognized and studied. Even some computer security jargon has its origin in biology. For example, in 1987, Adelman introduced the term ''computer virus'' (Cohen, 1987), which Spafford also depicted as ''a form of artificial life'' in 1992 (Spafford, 1992). The term ''computer worm'' first appeared in computer research in 1982 (Shoch and Hupp, 1982), and even reaches back to the science fiction novel *The Shockwave Rider* by John Brunner in 1975. But as we pointed out in Section 1, most work has focused on leveraging biological protection mechanisms for computer defense technologies.

The analogy between the protection mechanisms of living organisms and the security of computers and computer networks is indeed appealing. In 1995, Kephart et al. introduced a neural network virus detector to distinguish between programs infected by computer viruses and those uninfected. As biology has taught us that random mutations protect populations of biological organisms from the devastation of epidemics, in order to make the computer code less susceptible to security attacks, there has also been idea of introducing similar random mutations within code to create more heterogeneous computing environments (Forrest et al., 1997a). Recently, Knapp et al. (2003) explored the usage of cell biology as a reference discipline for network and information security. They specifically examined the similarity of a cell's defense mechanism to the defense of a networked computer system. Goel and Bush (2004) further pointed out that three biological mechanisms in cellular organisms are useful to develop security models for computer networks: genomics (RNA interference) for developing the defensive computer code that turns off the dangerous code, proteomics (protein pathway mapping) for mapping security events with specific networking paths, and physiology (immune system) for generating ''antibodies'' within a computer system.

The most popular paradigm in drawing security lessons from the biological world is probably the application of natural immune systems in computer world. The immune system offers great insights in effectively distinguishing *self* and *non-self*, in being naturally resilient and adaptive to various forms of old and new pathogens, and in being efficient and fast. The earliest works in this direction are perhaps Forrest et al.'s (1994) file integrity verification method based on the generation of T-cells in the immune system and Kephart et al.'s (1995) computer immune system for identifying and removing computer viruses. Later more generic computer defense systems were studied. Forrest et al. (1997b) presented a computer defense system that was directly modeled after features from natural immune systems, such as multi-layered protection against foreign materials, distributed detection of non-self,

unique detection mechanism for every individual, detection of the previously unseen, and imperfect detection. Skormin et al. (2001) suggested that an information security system must include semi-autonomous security agents that adopt principles from biological immune systems. Hofmeyr in his Ph.D. dissertation explored an immunological model of distributed detection of network intrusions (Hofmeyr, 1999). Boukerche et al. (2004) proposed an artificial immune based intrusion detection model for computer and telecommunication systems. Boudec and Sarafijanovic (2004) even developed an artificial immune system just for misbehavior detection in mobile ad hoc networks.

3. Approach

The key to drawing useful correlations between biological and digital pathogens is the proper selection of comparison scope. Close comparisons between the physical mechanisms of infection are bound to fail due to the fundamental differences in construction between biological and digital systems. On the other hand, looking only at broad environmental issues misses infection strategies that have proven effective despite deliberately hostile host environments. Instead, we identify whether a biological infection is successful because of the environment or in spite of it, then draw conclusions about general trends that would carry over to the study of computer worms. We look at both (1) the efficacy of the pathogen's infection technique on the individual host, and (2) the environmental conditions that favor greater infection success. Note that in Sections 4 and 5 we present only our observations on the similarities between biological and computer infections, refraining from presenting our insights and conclusions until Section 6.

We divide our observations of biological infection strategies into *operational exploits* and *environmental exploits*. Operational exploits have proved evolutionarily successful on the merits of their individual techniques. Environmental exploits rely on conditions that have allowed for the spread of diseases. In a similar way to biological epidemics, digital outbreaks can be viewed as not just the result of bugs in specific programs, but also the result of loose security policy in systems and networks.

We study six of the most common fatal infectious diseases as reported by the World Health Organization (2002): measles, malaria, HIV/AIDS, tuberculosis, acute respiratory infections (for which we will study influenza as a representative disease) and (collectively) the diarrhoeal diseases. Each of the previous diseases represents a combination of operational and environmental exploits used to infect and spread in a biological entity and among a population. While the media attention is often drawn to the diseases that are caused by new and relatively rare biological viruses (such as the recent SARS (Centers for Disease Control and Prevention, 2005) or Asian bird flu (Centers for Disease Control and Prevention, 2006a)), it is these six contagious diseases that account for 90% of the deaths from communicable disease (World Health Organization, 1999). These diseases are also among the biggest disablers; according to World Health Organization (1999), "at any one time, hundreds of millions of people—mainly in developing countries—are disabled by infectious diseases." Understanding them and the defense mechanisms against them should offer insights on the propagation of digital diseases and defenses against them.

We could have also studied the most common infections regardless of severity, but the main threat from digital pathogens is from those that have the most destructive pattern. Head lice and adware may be annoying and even in extreme cases lead to complications, but they are not as serious threat as a life threatening illness is to human health or a lethal computer worm is to your data.

4. Operational exploits

Between biological diseases and computer worms, some of the most intriguing similarities lie in the ways in which they infect their hosts and bypass the defenses designed to stop them. All of the six diseases we examine have their own unique operational strategy to bypass or subvert a body's concerted efforts in either blocking the entrance of diseases or defeating them after infection. These operational strategies have close parallels to the behavior of computer worms.

To concentrate on the unique behavior of each contagion and how those behaviors appear again in computer worms, we will focus on techniques that abstract out the biological mechanics. In this section, we will explore the techniques of the most deadly biological diseases and some of the equivalent mechanisms in computer infections. Note that we exclude measles from our operation analysis, since the most prominent feature of measles is primarily related to environmental factors that we will discuss in Section 5.3.

4.1. *Malaria*

4.1.1. *Spread through a third party*

It is estimated that malaria causes around 20% of all deaths in children under five in sub-Saharan Africa (World Health Organization, 2005a). The destructive impact of the disease is due to not only ferocity of the disease itself, but also the mechanisms used in propagation. Malaria primarily relies on transport through mosquitoes (which are unaffected by the disease) (Centers for Disease Control and Prevention, 2004). A mosquito takes blood from an infected victim containing the disease, and then transfers some of the infected blood to the next person it bites. Because mosquito bites are a common occurrence, malaria is only treated after the appearance of symptoms. The actual infecting bite is often ignored because of the number of innocuous bites before it.

Computer applications can be exploited by worms to act as a third party to conceal the malicious infections of worms. For example, an email-borne worm that hides itself inside a specially crafted attachment of an email would have much higher chance of the recipient opening the attachment than if it transmitted with its own generic message (Computer Emergency Response Team, 2003a). Or, a worm could use infected web-servers as a platform for exploiting holes in visiting web browsers, which in turn could try to infect the servers they later visit (Staniford et al., 2002). The Nimda

worm (Computer Emergency Response Team, 2001a) even leverages both mechanisms (email and web browsers) in propagating through third-parties. By piggybacking on legitimate traffic, a worm can become very hard to detect without a specific signature.

4.1.2. *Use topographic spread patterns*

Another advantage that a disease like malaria gains from the mosquito is the distribution pattern. Outbreaks of traditional human-to-human infections can be contained using a simple quarantine system. The mosquito is an ubiquitous presence in infected regions, however, meaning that a simple quarantine will not be effective. We could reduce the overall number of mosquitoes, but they cannot feasibly be eliminated. While human interaction can be restricted, the mosquitoes follow their own travel pattern, carrying the disease to areas an infected person would have never visited.

Biological diseases sometimes tend to follow topographic spread patterns, following along roads and rivers with commerce and travel (Wilson, 1995; Montoya, 2004). Malaria's spread pattern follows the distribution of the mosquitoes as well as through human migration. The introduction of infected humans into an area can lead to the creation of a population of carrying mosquitoes, which can then spread the disease even after quarantining the originally infected individual. A diversity of transmission mechanisms makes for a persistence that defies simple quarantines.

Topographic spread is not a new idea in worm design (Staniford et al., 2002). Although so far most worms use a probabilistic mechanism to determine the next target of infection (such as by randomly generating IP addresses), this type of mechanism has low hit percentages and often produce characteristic traffic patterns that can be detected by a fairly simple process. It is easily foreseeable that when worms seek deeper stealth, faster speed, and a higher ratio of infection to attempts, they will become more aggressive in using information gathered from computer connection histories to more effectively select new targets of infection and piggyback on legitimate traffic (Staniford et al., 2002). Such aggressiveness has been proved by worms that have used local scanning preference (such as CodeRed II (Computer Emergency Response Team, 2001b)) and simulations predict ominous results from more sophisticated techniques (Zhou et al., 2005; Yu et al., 2005). Already E-mail worms commonly dredge mail agent files and hard drives to look for addresses that can be used to continue the infection (Computer Emergency Response Team, 2003a). A web-server worm could just as easily find other web servers by looking through the links on the pages the web server provides. Caches and application histories provide links to social networks of users with similar behavior and software, making them another source of topology information for an aspiring worm writer.

4.2. *Tuberculosis (TB)*

4.2.1. *Become more stealthy through long incubation periods*

Tuberculosis (TB) has always had the reputation of being a slow killer. Before the advent of modern antibiotics, it was common to see bouts of illness followed by dormancy for years on end (ten Asbroek et al., 1999). The time between initial infection and the onset of symptoms is also called an ''incubation period.'' An incubation period was needed for the disease to reach a population capable of producing ill effects, and its length may vary for victims at different health level. Because symptoms were already indicative of a serious infection, the prognosis was grim in the days before penicillin.

In computer-based infections, a worm author can doom the progress of a worm by making it too aggressive. As detection systems become better at spotting the characteristics of fast-spreading worms, worm authors may concentrate on stealth, compromising speed to maximize overall damage. A worm that exhibits a primitive stealth mechanism is the CodeRedII worm (Moore et al., 2002), which after infecting a victim will stay dormant for 24 hours. This quiescent period between the infection and the subsequent scanning activity makes it harder to find the original infecting connection in activity logs, and allows more hosts to become infected within a network before their scanning activity reveals that the network has been compromised.

Meanwhile, worm detectors today cannot capture all worms. If a worm detector finds worm patterns or worm-like system behavior in a program code, system analysts will receive warning signals and begin analyzing the suspicious code. If necessary, a signature will be generated and distributed to worm scanners. However, worm detectors often determine the occurrence of a worm based on the frequency of suspicious connections or byte patterns. If worm reduces the frequency of its connections, it may be able to slip by these detectors in the background noise. As long as the detectors do not raise the alarm and no one notices odd behavior, the worm will be invisible. It is possible that no one would know of its existence at all until it began to cause damage. Actions taken to defend against the infection would not impact the spread of the worm. If the malicious programs can gain adequate time to build up their strength by stealthy means before executing their payload, they can cause the most devastating damage!

4.3. *Influenza*

4.3.1. *Keep changing profile, adapt quickly*

Influenza (flu) is a contagious infection of the respiratory system caused by influenza viruses. Deadly complications from Influenza are most common in the elderly and small children, but they threaten anyone with a weakened immune system (Centers for Disease Control and Prevention, 2006b). The longevity and veracity of Influenza is closely related to the speed with which the virus changes its protein structure, which can make previously developed antibodies less effective (Ghedin et al., 2005). The pandemic of 1918 is a good example of how Influenza rapidly shifts its structure and makes people already possessing a certain level of immunity vulnerable again. Shifting structure invalidates vaccines based on previous structures, making a complete eradication scheme infeasible.

This biological phenomenon has a digital analog in polymorphic computer worms. When propagating, this kind of worm can change its own data or instructions while still

performing the same tasks so that worm scanners cannot recognize them (Weaver et al., 2003). For example, a worm can encrypt its own payload, and can produce different encrypted payloads by using different keys.

Polymorphic behavior can also be added to a worm after its release, fixing flaws in propagation and reacting to the signatures built to defend against it. For example, worms can use a command distribution channel to send out updated versions of themselves (Staniford et al., 2002). Whereas most host-level detectors use byte signatures to detect the malicious code on the computer, to get around signature scans the worm author could regularly distribute permutations of the code that do not match the signature but perform the same tasks. Also, different versions of worm code could be downloaded from the network.

4.4. *Diarrhoeal diseases*

4.4.1. *Deprive resources needed to fight disease*

The most common fatal complication arising from diarrhoeal diseases (such as cholera) is dehydration due to rapid fluid loss (World Health Organization, 2005b). The disease rapidly drains fluids from the victim, leaving them dehydrated and malnourished. The effects of this process become deadly in regions where there is already a high level of malnutrition and a lack of potable water. Cholera and related diseases do not directly attack the immune system like HIV—they achieve an indirect victory by depriving the body of the *resources* that are needed to fight the infection and maintain normal bodily operations. While a person can live several weeks without food, they can only live a few days without water. The diarrhoeal diseases are effective because by cutting off the water supply to the body, they deprive the defense of the time and resources it needs to destroy the invader.

Malicious computer worms can use strategically placed infections to deny defensive systems of the bandwidth, CPU time, or other resources that are necessary to respond aggressively to attacks. On a host level basis, for example, a worm could consume enough memory and processor time to slow defensive measures down to a crawl (Spafford, 1989). A worm can also create a zombie network or ''botnet'' composed of compromised machines (Geer, 2005), which can be requested by the worm to launch DDoS attacks against a centralized security server or the links that the server is using.

4.5. *HIV/AIDS*

4.5.1. *Attack the defense*

There are roughly 40.3 million people in the world infected with HIV (Joint United Nations Programme on HIV/AIDS, 2005). This number made all the more devastating by the fact that this growth has happened within only the last 26 years. Although the reasons for the massive increase of HIV infections are a complex mixture of behavioral, sociological and political trends, the reason for its 100% mortality rate lies in its behavior inside the human body. HIV infects and replicates inside a human body's T-cells while destroying these cells (Centers for Disease Control and Prevention, 2007). These are the very cells tasked with destroying infected cells and foreign invaders. As the infection grows and more T-cells are infected and destroyed, the body becomes less equipped to handle the virus. Eventually, the most important defenses of the human body are stripped, leaving the victim at the mercy of the pathogens that are constantly around them.

Computer defense solutions that desire to become the equivalent of the human immune system may be targeted by similar types of attacks (Keizer, 2005). For example, the Win32/Blaster worm can launch a TCP SYN flood denial-of-service attack against windowsupdate.com, a Microsoft site that is in charge of managing software patches and security updates (Computer Emergency Response Team, 2003b). A subverted security program provides the user with a false sense of security while giving the malicious program automatic legitimacy in the eyes of the system. Security programs can also become denial-of-service agents when infected, sending out fake signatures and warnings that can cause uninfected machines to restrict legitimate operations and files.

4.5.2. *Let other vectors do the dirty work*

While HIV is the real killer of those that die from AIDS, it is always another infection that actually finishes the job. Once striped of defenses by the HIV virus, an afflicted person can develop a fatal case of pneumonia from an infection that normally would not advance to the stage of showing symptoms (Cohn, 1991). The population of the HIV viruses inside the infected person does not benefit from the fatal infections that it enables; yet this pattern of behavior conceals the real cause of the infection from the doctors treating it for some time.

A computer worm can perform a similar trick. With analogical analysis, we can deduce straightforwardly that a computer worm can choose to outsource the destructive operations to other vectors in order to escape detection. The goal is to fool not only the users on the infected hosts, but also the security community that is actively working against them. Imagine a worm that removes itself after infecting victim machines through an unknown vulnerability: It can open up a second (new or previously patched) vulnerability on victims and start a second worm to launch a DDoS attack or destroy data. Even if the second worm is caught, the original worm could make several successful runs before the real cause of the infection was discovered. For example, after the CodeRedII worm compromises a machine, it installs a ''backdoor'' at the machine to allow the attacker to remotely execute any arbitrary exploit in the future (Moore et al., 2002).

5. Environmental exploits

The threat posed by an infectious disease is determined by not only the way in which it spreads, but also where it spreads. Diseases that are all but eliminated in the developed world still persist in the third world where insufficient sanitation, poor health care, and malnutrition are fairly common (World Health Organization, 2002).

Furthermore, the fight against disease is not simply a matter of medical advancement. Even with vaccines and successful treatments available, certain diseases still claim large

numbers of victims. Treatments that have proven effective in the lab must be distributed to those at risk systematically and effectively. Noticeably, obstacles to the effective distribution of treatments are often environmental factors, instead of intrinsic reasons (World Health Organization, 2002).

In the digital world, networks and desktop users with insecure practices often provide breeding grounds for computer worms. In this section, we explore the environmental conditions that allow worms to flourish and the techniques that worms can use to exploit these conditions. We draw comparisons specifically with diarrhoeal diseases, tuberculosis, and measles, because they provide the most prominent examples of diseases whose success is largely due to the environment in which they operate. HIV, malaria and influenza all have environmental components that help their spread, but as these components provide no additional major insight into the behavior and spread of computer worms, we omit the discussion of these three diseases here.

5.1. *Diarrhoeal diseases*

5.1.1. *Infect aggressively in an unsanitary environment*

Diarrhoeal diseases are spread through the food and water supply and occur most widely in areas where sanitation is poor (World Health Organization, 2005b). While not all are serious by themselves, the constant assault of minor attacks against the immune system in an unsanitary environment slowly wears down the body's defenses. A combination of inadequate waste disposal and an impure water supply creates an environment hospitable enough for these diseases to disarm all but the healthiest individuals, and at that point they can become deadly.

Sanitation in the computer world is a more abstract concept describing the security measures in place on any local collection of computers. A computer that is collocated with many other machines that are vulnerable or even already compromised is not in a sanitary environment. Also, a computer or a network of computers that is exposed to all kinds of inbound malicious probes without a firewall to fend off outside attacks, for example, is located in a ''dirty'' environment (Cheswick et al., 2003). Worm attacks are generally more successful in attacking these computers than those under strengthened security protection.

A particular kind of worm that exploits the local pollution within a local area network is the *local preference worm* (Staniford et al., 2002). When propagating from an infected host, this kind of worm prefers to scan and infect the other hosts in the same local area network. Local preference scanning worms are a common occurrence these days and can produce far more local network infections in a shorter amount of time than standard random scanning behavior.

5.2. *Tuberculosis (TB)*

5.2.1. *Attack targets that are under unhealthy conditions*

The severity of an infection is not only related to the pathogen doing the infecting, but also dependent on the health conditions of the host being infected. Relatively minor infections become life threatening when combined with malnourishment, fluid loss or a weak immune system. Tuberculosis (TB) strikes at people who have already been assaulted by hostile environmental conditions around them (National Institute of Allergy and Infectious Diseases, 2006); previous infections and impure water may further weaken the body's defenses against TB.

Computer worms also succeed first in infecting those hosts that are not under ''healthy'' conditions (Weaver et al., 2003). Often, worms can most successfully infect machines running buggy, unsteady operating systems or applications, machines without effective resource access control mechanisms, or machines ignoring ''least privilege'' or other well-established security principles (the least privilege principle requires that a program can only have privileges immediately needed for accessing resources in a system (Saltzer and Schroeder, 1975)). Furthermore, if a worm can launch on an unhealthy host using one of its particular vulnerabilities, the worm could further survey the host for other vulnerabilities or introduce their own. This phenomenon suggests that computer systems should not only run security software to protect themselves from worms, but should also position themselves in a ''healthy'' condition.

The ''unhealthiness'' of a computer could also relate to the behavior of users of the computer. For instance, the appeal of desktop systems as vectors for infection is not just in the size of the vulnerable population, but also in the behavior of the user. While corporate or institutional servers are hard targets with small populations and a significant chance of detecting worms, desktops on the other hand tend to be poorly defended and run by unsophisticated users. Lax security policy by network administrators can further expose hosts to threats from both outside and inside the network (Cheswick et al., 2003). These vulnerable networks and computers compose the dark corners of the Internet which can be used to springboard more ambitious worms.

5.2.2. *Exploit environmental factors to continuously evolve*

As antibiotic treatments are used against TB, the ''fittest'' TB strains that are resistant to antibiotics being used can survive and even thrive. Also called ''selection'' in biological terms, this evolution process has led to an increased prevalence of TB's antibiotic resistance (National Institute of Allergy and Infectious Diseases, 2006). Moreover, human and social factors contributed to the resistance. For example, not finishing the TB medicine can allow TB strains that are resistant to standard TB drugs to survive, leaving the patient still ill (Lewis, 1995). Also, the perception (especially in the 1980s) that TB bacteria could be killed by a number of commonly used antibiotics led to an overuse of these antibiotics, causing TB bacteria to increase their resistance against these antibiotics. Clearly, only relying on those commonly used antibiotics can create a monoculture that allows TB to quickly outwit the TB antibiotics through evolution.

Computer worms have often outwitted defense mechanisms by evolving themselves. One such example is the Sobig worm family (Computer Emergency Response Team, 2003a). While worm detection software continues to learn the individual fingerprints of worms, zero-day worms also continue to appear. In particular, when computers all rely

on just one or a small number of uniform mechanisms to defend themselves against worms, perhaps in order to minimize performance penalty or ease computer administration, it would create a monoculture in defense (Goth, 2003), and a single flaw would lead to a severe breakout of new worms.

5.3. *Measles*

5.3.1. *Aim at the population not covered by vaccines*

The vaccine for measles has been around since 1963, yet complications from the disease are estimated to still take 875,000 lives a year in developing countries; that accounts for over 50% of the deaths caused by vaccine-preventable diseases (World Health Organization, 2003). While part of the problem has been the extreme communicability of measles (measles is an air-borne pathogen and spreads rapidly throughout a household), the major cause is primarily attributable to the under-utilization of measles vaccine.

Similarly, the code for computer worms is as widespread as the copies of unpatched software that worms exploit. Unpatched systems extend the destructive behavior of computer worms, well past the point where technical solutions are available. Paxon, Weaver and Staniford also point out that, because new machines are installed with old versions of software that contain vulnerabilities, more computers are infected by computer worms (Staniford et al., 2002)—the same as measles taking advantage of the growth of vulnerable populations to continue its existence. While older people (or existing computer systems) may already have immunity to the disease (or worms), an unvaccinated (or unpatched) generation can provide an active breeding ground to pass the disease (or worms) on.

5.3.2. *Take advantage of complacency for more damage*

Perhaps the most significant environmental factor in disease propagation is complacency. While diseases like smallpox garnered widespread support for eradication because of their high mortality rate even amongst the strong and healthy, diseases with low mortality rates are sometimes tolerated even in the most developed countries (Immunization Action Coalition, 2002). Becoming infected with measles or chicken pox is even considered a common part of growing up. Unfortunately, by accepting some level of infection because of the mild symptoms of the common case, we provide the necessary complacency for diseases to continue their propagation. Measles is a rather mild disease under ideal conditions, but can be fatal, and by allowing it to persist we expose ourselves to some risk, however small it may be.

A similar phenomenon exists in the computer world. A common desktop user may simply view his/her computer as a personal tool, without considering the broader ramifications of network connectivity. Once that computer is connected to the Internet, however, it has become part of a digital community. This sense of joining a community is less concrete than the physical analogy of moving to a new neighborhood, but it has similar consequences. Once a user's machine becomes connected to others, it immediately faces a variety of threats from the network (Cheswick et al., 2003). Additionally, if it is not well protected, the machine may also become a threat to the other machines on the network.

Furthermore, vigilance against computer threats is often tied to the perceived severity of these threats. A user that primarily uses his/her computer for web browsing and communicating with friends might see little need for security. Likewise, a network administrator for a small auto parts distributor may see little threat from malicious attack due to the nature of his/her business. But as long as the consequences of infection are less than catastrophic to an average user, complacency tends to remain high and the risk to all the members of the digital community is increased.

6. Insights on worm defense

We can draw insights on worm defense based on both the operational and environmental exploits characteristic of biological diseases. Based on Sections 4 and 5, we elaborate our six main insights in this concluding section.

6.1. *Worm traffic can only be stopped when distinguishable from legitimate traffic*

Sections 4.1.1–4.3.1 show that worms can piggyback themselves on legitimate traffic to propagate; moreover, they can explore topology information, incubate, or become polymorphic to make them indistinguishable from legitimate traffic. To stop worm traffic from spreading, one must be able to understand and find the unique, essential characteristics of worm traffic before knowing how to stop them. For example, byte patterns used as worm signature often fail to identify worm traffic when worms change their payload. This insight has led researchers in new directions in search of effective detection of *zero-day* worms, as incorporated by Li et al. (2006) in their SWORD worm detector.

6.2. *Computer resources and computer defense systems are targets for infection and deception*

In Sections 4.4.1–4.5.2 we have found that computer worms can target computer resources and computer defense systems in order to spread more effectively. This affirms the point that the security of a system hinges heavily on the protection of the resources in that system. It is not only about ensuring that the resources *are not* available to those that do not need to access the resources, but also about ensuring that the resources *are* available to those that do. Moreover, computer defense systems themselves must be strongly secured. Security systems should not be a license for complacency or an excuse to allow vulnerable behavior. Like any other application on the network or system, security applications should be treated with an air of distrust and monitored for anomalous behavior. Enough worm writers have realized the threat of common security programs, and attempt to disable the most common ones from running or prevent them from running properly. Anti-virus, firewall and intrusion detection systems are still complex software systems, offering opportunities for an unchecked buffer or a forgotten testing routine. The potential damage that can be caused by subverting systems with

security software is not only in the numbers of computers that can be infected, but also in the length of time that it can operate undetected.

6.3. *Enforce comprehensive "sanitation"*

Sections 5.1.1–5.2.1 show that when a system or its external environment is not clean and healthy, it can be much easier for computer worms to penetrate. Digital "sanitation" is therefore necessary to control and filter what comes in and goes out to try to create a sterile environment inside a system or network. However, while a perfect filter can protect against all infections except those developed on the machine itself, in practice, filters are only as good as the rules they have to go by. Those rules are usually created in response to an infection that has already happened, and a realistic filtering scheme should seek to exclude any suspicious-looking traffic that displays certain behavior patterns, contains questionable content, or utilizes an unauthorized service.

The level of sanitation that can be achieved also depends on the amount of control the defender has over the system to protect and the extent to which they utilize that control. All systems have to make a tradeoff between security needs and user needs. A system that rigorously controls all user behavior can be more secure but may not be flexible enough to suit the needs of the user. A home user can rigorously filter everything, but an ISP that tries to dictate the behavior of its subscribers will drive away customers. On the other hand, today the threat is constant and serious. The level of sanitation needed need to be adequate to fend off infection.

6.4. *Defend in depth with diversity*

Section 5.2.2 shows that computer worms will continue to evolve. Zero-day worms will appear from time to time. In addition to finding unique, essential characteristics of worm traffic in order to stop them (as described earlier in Section 6.1), defending a system in depth with sufficient diversity is also essential. Security should never rest on the assumption of correctness of a single mechanism; a diverse set of detection applications and responses improves the overall health of a network (such as the Internet) at large. Any single security mechanism can eventually be breached by a focused effort; a diversity of these mechanisms prevents the effort from being automated. This is not to suggest that every workstation needs to be loaded down with an array of redundant defense mechanisms. It is important, however, to consider value of diversity in protection against the desire for a single, centralized solution.

6.5. *Patch computer systems proactively, but still assume a hostile environment on startup*

Section 5.3.1 shows that unpatched systems create playgrounds for worms. While effective means are required to stop worm traffic, to completely eliminate worms, they must be denied refuge at vulnerable hosts through which they can spread. In particular, one must be vigilant in ensuring that an effective, comprehensive and timely software patching system, such as that proposed in Li et al. (2004), is in place to deny worms the software holes they need to propagate, just as a comprehensive vaccination program must be in place to eradicate measles. Incomplete patching has not been effective in counteracting computer worms, as exemplified by the initial large-scale outbreak of the CodeRed worm and the monthly resurgence thereafter, despite a patch being readily available (Staniford et al., 2002).

Unfortunately, comprehensive, timely patching is not easy to achieve in reality, and we must assume a hostile environment on startup. As in the case of providing a measles vaccine to all children worldwide, it has been impossible to ensure *all* computers are patched in order to be resilient against computer worms, and it probably will remain this way at least in the near future. On the other hand, all software of moderate complexity has bugs, for security's sake we must assume that all these bugs could grant complete control of the system. We must also assume that at the time of installation, these bugs are known and there are worm programs actively searching for unpatched machines with these bugs in order to install worm code. An animal born without an immune system of its own would rapidly fall prey to the infectious diseases all around them at the time of birth. An application not patched or not designed to actively fend off threats on the initial startup will soon fall victim to its out-of-date code.

Removing the exploit that worms use to propagate is the only long-term way to eliminate their impact. The case can be made for an application or system to start up with the bare minimum of functionality and only become fully operational when it receives the most recent patches from its designer. As high-speed Internet access becomes more common, it is not an unreasonable assumption that properly licensed users of an application will be able to connect to a set of central servers to receive updates. When no network connection is available the assumption of a hostile environment is no longer necessary and the application could be used unpatched. Combined with a mandatory periodic update policy that balances bandwidth requirements with the freshness of the code, safe startup could minimize damage from any single exploit. The question then is: under what conditions might a user accept a software license with mandatory updates and safe startup?

6.6. *Infections will still happen, be ready to respond*

Section 5.3.2 points out that user complacency could be the most significant factor allowing severe worm spread. As computer systems add new functions and become more complex, they are often harder and more costly to secure, which could lead to a higher level of user complacency. Administrators must reconcile themselves to the idea that every system they control has flaws that can be exploited. Instead of being an excuse for apathy, insecurity requires even greater diligence. Defense should focus as much on the reaction to infection as the avoidance of it. Detection of infection should be followed up by a strict quarantine; and host-level defensive mechanisms on the infected computer should be considered compromised.

REFERENCES

Boudec JL, Sarafijanovic S. An artificial immune system approach to misbehavior detection in mobile ad-hoc networks. In: Proceedings of Bio-ADIT 2004 (The First International Workshop on Biologically Inspired Approaches to Advanced Information Technology); January 2004. p. 96–111.

Boukerche A, Jucá KRL, Sobral JB, Notare MSMA. An artificial immune based intrusion detection model for computer and telecommunication systems. Parallel Computer 2004;30(5–6): 629–46.

Centers for Disease Control and Prevention. Key facts about avian influenza (bird flu) and avian influenza A (H5N1) virus; June 2006a. Available from: http://www.cdc.gov/flu/avian/gen-info/facts.htm.

Centers for Disease Control and Prevention. HIV/AIDS; January 2007. Available from: http://www.cdc.gov/hiv.

Centers for Disease Control and Prevention, Influenza (flu); August 2006b. Available from: http://www.cdc.gov/flu.

Centers for Disease Control and Prevention. Malaria; May 2004. Available from: http://www.cdc.gov/malaria.

Centers for Disease Control and Prevention. Severe acute respiratory syndrome (SARS); May 2005. Available from: http://www.cdc.gov/NCIDOD/SARS/.

Cheswick WR, Bellovin SM, Rubin AD. Firewalls and Internet Security; repelling the Wily Hacker. 2nd ed. Reading, MA: Addison-Wesley; 2003.

Cohen F. Computer viruses. Computer and Security 1987;6:22–35.

Cohn D. Bacterial pneumonia in persons infected with the human immunodeficiency virus. Infectious Disease Clinics of North America September 1991;5(3):485–507.

Computer Emergency Response Team. CERT advisory CA-2001-26 Nimda worm; September 2001a. Available from: http://www.cert.org/advisories/CA-2001-26.html.

Computer Emergency Response Team. CERT incident note IN-2001-09 Code Red II: another worm exploiting buffer overflow in IIS indexing service DLL; August 2001b. Available from: http://www.cert.org/incident_notes/IN-2001-09.html.

Computer Emergency Response Team. CERT incident note IN-2003-03 W32/SoBig.F worm; August 2003a. Available from: http://www.cert.org/incident_notes/IN-2003-03.html.

Computer Emergency Response Team. CERT advisory CA-2003-20 W32/Blaster worm; August 2003b. Available from: http://www.cert.org/advisories/CA-2003-20.html.

Forrest S, Perelson AS, Allen L, Cherukuri R. Self-nonself discrimination in a computer. In: Proceedings of the 1994 IEEE symposium on security and privacy. Washington, DC, USA: IEEE Computer Society; 1994. p. 202.

Forrest S, Somayaji A, Ackley DH. Building diverse computer systems. In: Workshop on hot topics in operating systems; 1997a. p. 67–72.

Forrest S, Hofmeyr SA, Somayaji A. Computer immunology. Communications of the ACM 1997b;40(10):88–96.

Geer D. Malicious bots threaten network security. IEEE Computer January 2005;38(1):18–20.

Ghedin E, Sengamalay NA, Shumway M, Zaborsky J, Feldblyum T, Subbu V, et al. Large-scale sequencing of human influenza reveals the dynamic nature of viral genome evolution. Nature. Available from: http://dx.doi.org/10.1038/nature04239. October 2005.

Goel S, Bush SF. Biological models of security for virus propagation in computer networks. LOGIN, December 2004;29(6): 49–56.

Goth G. Addressing the monoculture. IEEE Security and Privacy 2003;November–December:8–10.

Hofmeyr SA. An immunological model of distributed detection and its application to computer security. Ph.D. dissertation, University of New Mexico; May 1999 [Adviser: Stephanie Forrest].

Immunization Action Coalition. Complacency the likely cause of measles epidemic in the seattle area; October 2002. Available from: http://www.immunize.org/reports/report051.asp.

Joint United Nations Programme on HIV/AIDS. AIDS epidemic update 2005; December 2005. Available from: http://www.unaids.org/epi/2005/.

Keizer G. Symantec anti-virus software open to attack. In: TechWeb technology news; 20 December 2005. Available from: http://www.techweb.com/wire/security/175007129.

Kephart JO, White SR. Directed-graph epidemiological models of computer viruses. In: Proceedings of IEEE symposium on security and privacy; May 1991. p. 343–61.

Kephart JO, White SR. Measuring and modeling computer virus prevalence. In: Proceedings of the 1993 IEEE symposium on research in security and privacy; 1993. p. 2–15.

Kephart JO, Sorkin GB, Arnold WC, Chess DM, Tesauro GJ, White SR. Biologically inspired defenses against computer viruses. In: Proceedings of international joint conference on artificial intelligence; 1995. p. 985–96.

Knapp K, Morris F, Rainer Jr RK, Byrd TA. Defense mechanisms of biological cells: a framework for network security thinking. Communications of the Association for Information Systems 2003;12(47):701–19.

Lewis R. The rise of antibiotic-resistant infections; September 1995. Available from: http://www.fda.gov/fdac/features/795_antibio.html.

Li J, Reiher PL, Popek GJ. Resilient self-organizing overlay networks for security update delivery. IEEE Journal on Selected Areas in Communications, Special Issue on Service Overlay Networks January 2004;22(1):189–202.

Li J, Stafford S, Ehrenkranz T. SWORD: self-propagating worm observation and rapid detection, University of Oregon. Technical Report CIS-TR-2006-03; 2006.

Montoya ID. Topography as a contextual variable in infectious disease transmission. Clinical Laboratory Science Spring 2004.

Moore D, Shannon C, Claffy K. Code-red: a case study on the spread and victims of an Internet worm. In Proceedings of ACM Internet measurement workshop; 2002.

Murray WH. The application of epidemiology to computer viruses. Computer Security 1998;7:139–50.

National Institute of Allergy and Infectious Diseases. Tuberculosis; March 2006. Available from: http://www.niaid.nih.gov/factsheets/tb.htm.

Saltzer J, Schroeder M. The protection of information in computer systems. Proceedings of the IEEE September 1975;53(9): 1278–308.

Shoch JF, Hupp JA. The worm programs. Communications of the ACM 1982;25(3):172–80.

Skormin VA, Delgado-Frias JG, McGee DL, Giordano J, Popyack LJ, Gorodetski VI, et al. BASIS: a biological approach to system information security. In: MMM-ACNS '01: proceedings of the international workshop on information assurance in computer networks. London, UK: Springer-Verlag; 2001. p. 127–42.

Spafford E. The Internet worm: crisis and aftermath. Communications of the ACM June 1989;32(6):678–87.

Spafford EH. Computer viruses—a form of artificial life?. In: Langton CG, Taylor C, Farmer JD, Rasmussen S, editors. Artificial life II. Redwood City, CA: Addison-Wesley; 1992. p. 727–45.

Staniford S, Paxson V, Weaver N. How to own the Internet in your spare time. In: Proceedings of USENIX security symposium; August 2002.

ten Asbroek A, Borgdorff M, Nagelkerke N, Sebek M, Devillé W, van Embden J, et al. Estimation of serial interval and incubation period of tuberculosis using dna fingerprinting. The International Journal of Tuberculosis and Lung Disease May 1999;3(5):414–20.

Weaver N, Paxson V, Staniford S, Cunningham R. A taxonomy of computer worms. In: Proceedings of workshop on rapid malcode. New York, NY, USA: ACM Press; 2003. p. 11–8.

Williamson MM, Léveillé J. An epidemiological model of virus spread and cleanup 2003. Available from: http://www.hpl.hp.com/techreports/2003/HPL-2003-39.html.

Wilson ME. Travel and the emergence of infectious diseases. Emerging Infectious Diseases 1995;1:39–46.

World Health Organization. World Health Organization report on infectious disease—removing obstacles to healthy development. 1999. Available from: http://www.who.int/infectious-disease-report/index-rpt99.html.

World Health Organization. World Health Organization report on infectious diseases. April 2002. Available from: http://www.who.int/infectious-disease-report.

World Health Organization & United Nations Children's Fund. Measles: mortality reduction and regional elimination. Strategic plan 2001–2005; March 2003. Available from: http://www.who.int/vaccines-documents.

World Health Organization. Roll Back Malaria Department, Malaria control today: current WHO recommendations. Geneva, Switzerland; March 2005a. Available from: http://www.who.int/malaria/docs/MCT_workingpaper.pdf.

World Health Organization. The treatment of diarrhea—a manual for physicians and other senior health workers; 2005b. Available from: http://www.who.int/child-adolescent-health/New_Publications/CHILD_HEALTH/ISBN_92_4_159318_0.pdf.

Yu W, Boyer C, Chellappan S, Xuan D. Peer-to-peer system-based active worm attacks: modeling and analysis. In: Proceedings of IEEE international conference on communications (ICC); May 2005.

Zhou L, Zhang L, McSherry F, Immorlica N, Costa M, Chien S. A first look at peer-to-peer worms: threats and defenses. In: Proceedings of international workshop on peer-to-peer systems; 2005. p. 24–35.

Dr. Jun Li is an assistant professor at the University of Oregon, and directs the Network Security Research Laboratory there. He received his Ph.D. from UCLA in 2002 (with honors), M.E. from Chinese Academy of Sciences (ISCAS) in 1995, and B.S. from Peking University in 1992. His current research includes Internet worm detection, Internet routing forensics, Internet IP source address validity, and security for peer-to-peer networking.

Paul Knickerbocker is a master student at the University of Oregon, working with Prof. Jun Li on Internet worm defense. He received his B.S. from the University of Oregon in 2005. His research interests are in Internet security and peer-to-peer networking.