A PRoactive Malware Identification System based on the Computer Hygiene Principles[*]

Vasileios Vlachos, and Diomidis Spinellis

Department of Management Science and Technology,
Athens University of Economics and Business (AUEB),
Patission 76, GR-104 34, Athens, Greece
email: {vbill,dds}@aueb.gr

Abstract

Recent worm epidemics have proven beyond any doubt that the existing centralized worm containment mechanisms are no longer adequate to protect vulnerable systems, resulting in a shift towards distributed cooperative mechanisms that aim to safeguard and immunize the susceptible population. We are presenting PROMIS, a P2P based algorithm that provides its participants with early information regarding the existence of a worm epidemic and allows them to automatically adjust their security level. Our argument is that our approach is based on the principles of hygiene: taking the basic precautions to avoid infection when an epidemic is on the rise and no cure is available.

The policy of being too cautious is the greatest risk of all
—Jawaharlal Nehru (1889 - 1964)

Keywords

Peer to Peer Networks, Computer Worms, Computer Hygiene

1 Introduction

Latest malware incidents (Moore *et al.*, 2002; Zou *et al.*, 2002; Bailey *et al.*, 2005) demonstrated the enormous harm that modern worms can cause, but also showed that new breeds of malcode appear almost concurrently with the announcement of new vulnerabilities (Shannon & Moore, 2004) or even before them (a situation known as *zero day vulnerability exploitation*). Thus it is clear that traditional security applications such as firewalls, IDS or anti-viruses, while beneficial, no longer provide sufficient protection against rapidly spreading malware. Therefore, it is useful to explore other more radical protective mechanisms that can act complementary to the existing security infrastructure. To achieve that, it is essential to understand the propagation dynamics of fast spreading worms so as to identify the available time frames, in which reaction is both feasible and effective. Staniford et al (Staniford *et al.*, 2002) proved that highly virulent worms are fully capable of infecting the susceptible population in less than 15 minutes, as in the case of a Warhol worm. Empirical evidence confirms the validity of these assumptions: the Slammer worm (Moore *et al.*, 2003a) required only 10 minutes to infect the vulnerable population using the simplest propagation strategy (random scanning), while the theoretical limits of an ultra virulent worm fall well below the psychological one minute limit (Weaver *et al.*, 2004; Staniford *et al.*, 2004).

To alleviate the effects of rapid malcode we propose a cooperative containment algorithm based on the following assumptions.

[*] *Information Management and Computer Security*, 15(4):295–312, 2007.

1. The most vulnerable systems are personal computers whose respective owners do not have the time nor the skills to protect them sufficiently. In other words their owners are not security professionals that are constantly aware of the trends of malware activity. At best it is reasonable to expect the aforementioned to:
   (a) Have enabled the automatic downloading and installation of the new updates and patches on their OS of choice (usually some flavor of Windows).
   (b) Have installed some kind of security application. Most probably an anti-virus with automatic updating of signatures and a roughly configured firewall. Experienced users or small offices with a minimal IT infrastructure may also host an Intrusion Detection System.
2. Fine grained security policies are perfectly suited for large organizations and enterprises that have valuable digital assets to protect in very complex environments with numerous different user groups, each one requiring different resources and access rights to perform their duty. On the other hand, small office or home office (SOHO) users can be sufficiently protected even with less detailed security policies. We are arguing by simply changing different predefined security policies it is possible to hold the attacks of most malware at bay. As most attacks have common attack vectors, such as the HTML engine of a popular e-mail client or the scripting abilities of an equally popular browser, by disabling these services only during worm epidemics and re-enabling them after the containment of the epidemic, we can adequately protect them against most threats. SOHO users on the other hand, may agree with a temporary hardening of their system in order to protect themselves against a malcode epidemic, but they are unlikely to accept a permanent disabling of their useful but not essential services and applications.

Our algorithm is called PRoactive Malware Identification System — PROMIS and is based on a peer-to-peer (P2P) architecture to provide timely information to the members of a specially crafted P2P group.

2 Related Work

P2P networks are widely treated as a potential propagation vector for malicious software. While many worms or viruses utilized so far P2P networks to accelerate their propagation, mostly masquerading as pirated software or media files, conversely we believe that their distributed architecture can offer significant advantages over the traditional centralized or partially centralized architectures. Our research has been greatly influenced by Kephart et al (Kephart *et al.*, 1993; Kephart, 1992; Kephart & White, 1999), who introduced in his seminal work the concept of computer epidemiology by applying the basic epidemiological models to computer viruses.

This work shares more common ground with the Indra project (Janakiraman *et al.*, 2003) and the quarantine reputation-based system of Coull et al (Coull & Szymansky, 2005). Indra's philosophy is quite similar to ours, but with one major difference. Our goal is to give to all participants of the PROMIS system the rate of the ongoing malicious activity and allow them to decide for the best applicable measures for themselves, while the Indra project aims to inform all the participants about specific threats as well as the origin of these threats in order to have them blacklisted. Coull's architecture is also based on a P2P framework, but they work at a router level and each node acts to protect the community in general, whereas in our case each node aims to protect itself, which diminishes the possibilities of Byzantine situations where malevolent nodes try to use innocent nodes to harm others. The information provided by DSHIELD (DShield, 2007) in which numerous clients submit data from their firewalls and IDS, that are collected and analyzed by a central server to derive attack trends and rates, has proven itself to be very useful for us. We could say, metaphorically speaking, that our design is a fully decentralized DSHIELD in a more general and simple form. Another system that aggregates data from thousands of clients and extracts global attack rates to inform a special members group is the DeepSight (DeepSight, 2007) system, but as a commercial service it does not provide all the required information to enable proper evaluation. PROMIS was designed with two things in mind. First that the spread of the recent worms cannot be suppressed using traditional centralized containment techniques, thus a highly distributed environment based on the P2P networks might be useful, and second that the overwhelming majority of most users do not want the remote installation of any kind of code to their systems from anyone besides the original vendor of their software. Therefore we find all automatic immunization and vaccination systems (Goldenberg *et al.*, 2005) or the concept of *good worms* (Kim & Kang, 2004; Middleton, 2001) are not an applicable solution at present. Stricter measures such as the quarantine mechanisms via whitelists and blacklists (Moore *et al.*, 2003b) are too difficult to be implemented. Therefore, more applicable solutions utilizing multiple firewalls to contain rapid malcode in different segments of large Wide Area Networks (WAN) are already in use (Staniford, 2003). Briesemeister et al (Briesemeister & Porras, 2006; Briesemeister & Porras, 2005) have made significant contributions to the design of quarantine algorithms using game theoretic approaches and formal methods, though they did not disclose all the implementation details of a future system based on their work. On the other hand Keromytis et al present the COVERAGE algorithm, which takes most of the practical issues into account. The COVERAGE algorithm (Anagnostakis *et al.*, 2007) has a lot of common characteristics with our PROMIS

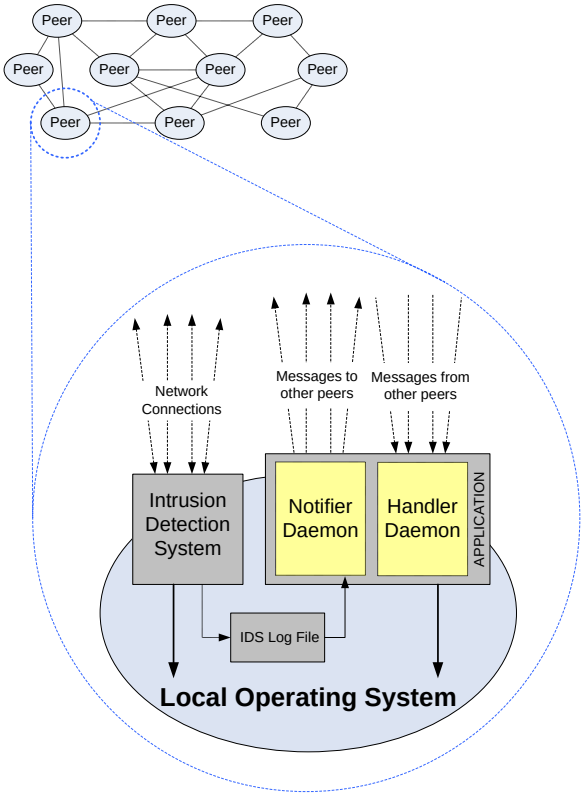

Figure 1: PROMIS architecture

algorithm, albeit their approach is by far more complicated as it aims to identify and categorize specific types of malcode. Since the cooperative prevention of rapid malcode is an open problem, we followed an approach which is similar to the principles of hygiene by taking the basic precautions when an epidemic is on the rise and letting others to come up with the most effective solution against particular types of attacks.

3 Architecture

PROMIS utilizes a P2P architecture (Figure 1). We assume that a special purpose security peer group, named PROMISGROUP, is created. PROMISGROUP contains two types of nodes, the *member* nodes and the *super* nodes. All normal nodes wishing to participate to this P2P group must authenticate themselves to one of the available super nodes. We consider that the authentication procedure takes place using secure out of bounds communication mechanisms. The super-nodes verify all the submitted data of a requesting node before authenticating it. This data may also include the real names, e-mail address and a phone/FAX numbers of the node's owner. Thus all member nodes of the PROMISGROUP are not intentionally malevolent, and more importantly, they can later be contacted if their behavior is unexpected or abnormal. The authentication procedure is outside the scope of our algorithm, as there are a number of excellent trust management schemes for P2P networks available in the literature (Androutsellis-Theotokis & Spinellis, 2004). We do provide, however some very simple mechanisms to exclude misbehaving nodes. We also require from every participating node to host a security application such as an anti-virus, firewall or an Intrusion Detection System in order to contribute to the more accurate estimation of the general malware activity, though it is possible for a node to gain from our system even if it doesn't operate any security application.

PROMIS nodes constantly perform two operations (Figure 2). A daemon called Notifier repeatedly checks during predefined short time intervals the log files of the security applications operating on the specific node and extracts the rate of the intercepted malicious activity against this host according to the following formula

$$p_t^n = \frac{h_t^n - \frac{\Sigma_{i=t-k}^{t-1} h_i^n}{k}}{\frac{\Sigma_{i=t-k}^{t-1} h_i^n}{k}} \quad (1)$$

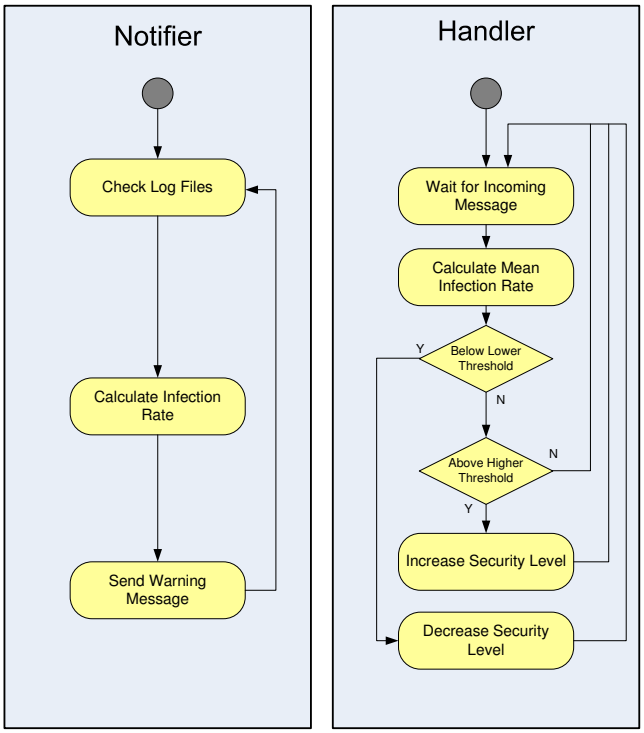

Figure 2: Handler and Notifier activity diagrams

where t is the ordinal number of a fixed time interval, n is the node identifier, h_t^n is the number of attacks node n received in the interval t, p_t^n is the percentage increase or decrease in attacks during the current interval t on node n, $k(>0)$ is the size of the 'time window' used in the number of t time intervals which the malicious activity rate is calculated. The Notifier also sends this local malicious activity rate to a number of randomly chosen participants of the PROMISGROUP.

Another daemon named Handler constantly listens for incoming rates from other peers of the PROMISGROUP and aggregates their messages in order to compute a global malicious activity rate for the PROMISGROUP using the following equation.

$$p_{avg} = \frac{\sum_{i=1}^{n} p_t^i}{n} \tag{2}$$

The Handler's main responsibility is to automatically adjust the security level of the local system based on the subsequent directives

- if $p_{avg} > r_{high}$, then increase the security policy by disabling non essential services as for example HTML preview in mail clients or by increasing the security settings of the installed web browser, where r_{high} is the predefined threshold to increase the security settings of the PROMIS system.
- if $p_{avg} < r_{low}$, then decrease the security policy by reactivating the above-mentioned services, where r_{low} is the predefined threshold to decrease the security settings of the PROMIS system.
- if $r_{low} \leq p_{avg} \leq r_{high}$ do nothing.

4 Implementation Details

The architecture described above is the core of the PROMIS algorithm. Before we investigated in detail the quantitative benefits of the PROMIS algorithm, we decided to build very small proof of concept prototypes so as to check whether the existent technologies are capable of supporting the PROMIS design. A PROMIS based system should be able to address all the issues that arise from a highly distributed environment. Moreover, its orientation towards cooperative security demands the use of mature technologies and robust infrastructures. Thereafter we identified

as the most common malware entry points the popular *I*nternet Explorer and *O*utlook Express applications which handle by default a number of important activities.

4.1 Netbiotic

Our first effort resulted in the *N*etbiotic project (Vlachos *et al.*, 2004). Netbiotic was written completely in Java and utilized the JXTA P2P framework from Sun in order to setup a testbed peer group in our lab. JXTA supports the establishment and operation of peer-to-peer networks. JXTA is a partially centralized p2p protocol implementation introduced in early 2001, designed for maximum peer autonomy and independence. It allows applications to be developed in any language, it is independent of any operating system type and is not limited to the TCP/IP protocol for data transfer.JXTA peers function under a role-based trust model, whereby individual peers function under the authority of third-party peers to carry out specific tasks. Public key encryption of the messages exchanged, which may be in XML format, as well as the use of signed certificates are supported, providing confidentiality to the system. The use of message digests provides data integrity, while the use of credentials - special tokens that authenticate a peer's permission to send a message to a specific endpoint - provide authentication and authorization. JXTA also supports the use of secure pipes based on the TLS protocol.

To test our concept we created a Netbiotic Peergroup in our lab. Thereafter all the PCs joined this peergroup and started to communicate. We simulated malicious activity by appending supposed malware attacks in various log files in at random and different rate for each system. Netbiotics' Notifier service in each peer monitors the changes in the supposed security log file and calculates the locally intercepted malicious activity. During fixed short time intervals it transmits the calculated rate to the Netbiotic Peergroup. Concurrently Netbiotics' Handler service collects the input from the other peers of the Netbiotic peergroup and periodically estimates the global malicious activity rate. If the global malicious rate exceeds the predefined threshold proper countermeasures are employed. In particular we utilized the Outwit tools (Spinellis, 2000) in order to automatically increase the security level of the Internet Explorer and the Outlook Express. Naturally if the estimated global malicious rate falls below the lower predefined threshold the security level of the above-mentioned applications is decreased.

The choice of JXTA as the underlying platform offers a great level of portability and platform independency, which is crucial in order to cover the largest possible set of systems and thus to maximize the benefits of Netbiotic. On the other hand some parts of the Netbiotic, such as the scripts that initialize the countermeasures, had to be platform specific.

4.2 MSPROMIS

A second approach was specifically targeted to homogeneous environments consisting solely of Microsoft's technologies, which are by far the most prevalent among end users without extensible security knowledge. Usually, these users are the preferred targets for most of the overall malicious activity, as they lack the technical skills to sufficiently protect themselves. MSPROMIS system (Vlachos *et al.*, 2005b) was based entirely on Microsoft's products. The development was completed in C# and C++. The P2P framework was established with the Microsoft Windows XP Peer-to-Peer Software Development Kit (Microsoft, 2003b). The Windows Peer-to-Peer Networking services require the installation of the IPv6 Protocol, which offers significant security enhancements over the traditional IPv4 protocol. The implementation of IPv6 from Microsoft (Microsoft, 2006) also necessitates the utilization of the Teredo transition technology (Microsoft, 2003a), so as to allow IPv6 connectivity across the IPv4 Internet. In our simplistic prototype some PCs were designated as supernodes of a newly formed MSPROMIS peergroup. After the formation of the PROMISGROUP, the peer members receive invitations via out of bounds means of communication to join the group. The invitation is a XML string that uniquely identifies the invited peer. Therefore, we are able to prohibit unauthorized access by malevolent peers and to preserve some control over the participating nodes. The architecture of the MSPROMIS system is identical to the architecture of the Netbiotic system, as both of them are alternative implementations of the PROMIS algorithm.

The default security application of the MSPROMIS is Microsoft's Internet Connection Firewall (ICF). The MSPROMIS Notifier utilizes the Microsoft's LogParser, a handy tool which allows the processing of various files via SQL queries. We employ the LogParser to pose various SQL queries in ICFs log files in order to extract the locally intercepted malicious activity. The MSPROMIS Handler service is very similar to the Netbiotic Handler, except for the fact that it is written in unmanaged C++ code. Finally, we reused the Outwit tool to change the security settings of the Internet Explorer and Outlook Express according to the estimated global malicious activity.

4.3 NGCE

Netbiotic and MSPROMIS were clearl proof-of-concept prototypes that helped us to conceptualize our ideas regarding the PROMIS algorithm. Furthermore, these two prototypes assisted us understanding the problems and issues that have to be addressed in order to deploy a large scale network of PROMIS systems. Moreover, the two

prototypes showed the technical feasibility of the PROMIS architecture. For reasons that will be evident in the next section we decided to measure the actual benefits of the PROMIS algorithms using simulations.

The most accurate results come from simulations that utilize real-data topologies, but because for the most part these data are not available (Paxson & Floyd, 1997), synthetic graphs are used instead. More importantly (Palmer & Steffan, 2000) showed the impact of graph generators in simulation outcomes, and (Tangmunarunkit *et al.*, 2002) pointed out the correlation between the physical infrastructure and the problem under investigation during the design of graph models. Therefore, we built the NGCE (Network Graphs for Computer Epidemiologists) tool that allows the generation of various graph models that are widely used in the computer viruses and worm propagation studies, as we reasoned in our previous work (Vlachos *et al.*, 2005a). NGCE is able to construct the following graph topologies:

- Homogenous graphs. Homogeneous or fully connected graphs were for many years the epidemiologists' preferred choice for describing the spread of infectious diseases. This topology has recently been adapted to model the growth of computer viruses and worms (Berk *et al.*, 2003; Kephart *et al.*, 1993; Kephart & White, 1999; Kephart, 1992; Chen *et al.*, 2003; Zou *et al.*, 2002; Staniford *et al.*, 2002; Staniford, 2003; Scandariato & Knight, 2004; Zou *et al.*, 2003b; Leveille, 2002). NGCE supports homogeneous graphs because they offer significant advantages. First, analytical mathematical models can be easily applied to them; second, they provide a good abstraction of very large networks when the majority of the susceptible hosts are accessible from an infectious agent and third, performing simulations using a homogeneous graph and comparing their outcomes with the mathematical analytical results is an excellent way to ensure that implementation details did not corrupt the simulation model.

- Scale-free graphs. Scale-free structures exist in a stunning range of heterogeneous systems ranging from biological and social to purely technological (Barabási & Bonabeau, 2003) networks such as the World Wide Web (WWW) (Kanovsky & Mazor, 2003; Adamic & Huberman, 2000; Bornholdt & Ebel, 2001), the physical connectivity of the Internet (Faloutsos *et al.*, 1999; Medina *et al.*, 2000) or the network of people connected by e-mail (Ebel *et al.*, 2002). Studies have explored the spread of malicious code in scale-free computer networks with interesting but conflicting results (Wang *et al.*, 2000; Leveille, 2002; Pastor-Satorras & Vespignani, 2001), while (Cohen *et al.*, 2000) investigating the resilience of the Internet in random breakdowns. We assume that, in light of this recent evidence, simulation research will increasingly be based on scale-free topologies and thus we support them in the NGCE tool.

- Random graphs. Until recently, it was believed that random graphs provide a good approximation to very large networks such as the Internet. Barabási et al (Barabási *et al.*, 1999) proved however, that the connectivity of the Internet, along with that of many other technical and social networks, obeys a power law distribution forming scale-free graphs. Prior to these findings, a large number of simulations that investigated the spread of malicious code, had been performed on random graphs (Kephart *et al.*, 1993; Kephart & White, 1999; Kephart, 1992; Zou *et al.*, 2003a; Leveille, 2002). Due to this fact, and, more importantly, in order to allow the comparison between older and current studies, we decided to include them in our tool.

- Lattices. Random graphs with fixed connectivity constitute a nontrivial network topology that is often encountered in grid systems. Considering the significant importance of such systems and the fact that older studies have been based on them (Kephart *et al.*, 1993; Kephart, 1992), we also included these structures in our NGCE tool

- Custom graphs. Sometimes it is necessary to measure the effects of various algorithms in non-standard graphs. We therefore added an option to our system that gives the ability to an experienced user to create non-typical graphs with custom properties based either on the random graph algorithm or on the scale-free algorithm.

NGCE can generate reproducible graphs (via a user-given seed) allowing the cooperation between various research groups as well as the validation of the experimental results of different scientists. We employed the NGCE tool to create a large test set of various graphs for our simulations. NGCE is written completely in Java and and is available via an open-source license at `http://ngce.sourcforge.net`. NGCE's design is modular, based on a general-purpose Graph class that provides the main functionality for all the graph operations such as adding, removing and counting edges. For each different graph topology a specific plug-in has been developed to implement the appropriate algorithm, making the development of new algorithms as separate plug-ins extremely easy. Various scripts calculate the probability distribution function $P(k)$, which gives the probability that a node has exactly k edges. After the completion of this step, $P(k)$ is automatically plotted using the gnuplot program. Every plot is displayed graphically, but is also stored in the Encapsulated PostScript (EPS) format. In addition, the title of each plot is created dynamically, in the form of an opaque Uniform Resource Identifier (URI) (Berners-Lee *et al.*, 1998). The title is adequate to provide all the necessary information of the plotted graph. Thus, every graph built and analyzed by NGCE can easily be reconstructed. Particularly in the case of random graphs, besides the experimental data, the expected theoretical distribution function is plotted as well. We believe that the

most important instrument to analyze the output graphs is the extraction of their statistical properties via NGCE's embedded scripts. On the other hand, we are aware that a visual representation of the generated graphs would provide additional means for researchers to decide whether a graph meets their needs. In order to add this type of functionality to NGCE, we took advantage of the popular Pajek (Batagelj & Mrvar, 2004) tool, which is able to draw graphs in a variety of different 2 and 3-dimensional plots. To accomplish this task, we made NGCE's output graph files compatible with the Pajek tool.

4.4 PROMISsim

In order to evaluate the PROMIS algorithm we developed the PROMIS simulator (PROMISsim) (Vlachos *et al.*, 2006). The PROMIS simulator is written completely in Java. This choice was made deliberately to utilize all the available powerful systems in our department with minimum effort. Indeed, we were able to perform our experiments seamlessly in our MAC OS X development workstations, the Windows systems of our colleagues and the FreeBSD and Linux servers of our laboratory.

To validate the correctness of the PROMIS simulator we performed a number of simulations of the unconstrained propagation of a worm. Subsequently, we compared the outcome of the simulation with the theoretical S-I (Susceptible-Infectious) or *s*imple epidemic model (Daley & Gani, 1999) which can also be calculated analytically. The following equations describe the propagation patterns of an unrestricted infectious agent in a homogeneous environment. Based on that model and by using the following differential equations, where N is the fixed population size, S is the number of the susceptible hosts, I is the number of the infected hosts, β is the *pairwise rate of infection* and under certain assumptions such as the homogeneous mixing of the population, it is possible to depict accurately the circulation of a disease.

$$\frac{dI(t)}{dt} = \beta * I(t) * S(t) \tag{3}$$

given that the population size is constant

$$N = S(t) + I(t) \tag{4}$$

PROMIS simulation results are sufficiently close to the analytical solutions of the S-I model. Thereafter we implemented the PROMIS algorithm and measured its ability to suppress the propagation of the rapid malcode. Initially we tested the full version of the PROMISsim application using very simple models. During that phase we calculated all the results manually, in order to gain confidence in our simulation model in its most complex version. Additionally we employed most of the commonly accepted technics (Law & Kelton, 2000) to avoid common problems (Andel & Yasinsac, 2006) in simulation based studies. Having performed numerous experiments with every factor that affects the outcome of the simulator, we saw consistency in our results as we will discuss in the following section.

5 Discussion

NetBiotic and MSPROMIS prototypes gave us the ability to gain valuable insights and prove the technical feasibility of our algorithm as well as to identify possible pitfalls and hazards. On the other hand the technical feasibility alone does not guarantee the practicability of the PROMIS architecture. Our lab experimental prototypes, although working to a certain degree, cannot by any means demonstrate the possible gains of the PROMIS architecture as a result of several reasons. The *in vitro* study of aggressive worms, even if all the necessary precautions have been taken to isolate the malcode from the rest of the networks, is still a dangerous act. Moreover these experiments could not take place on a university campus, in which it is quite difficult to enforce very strict security policies (Schneier, 2006).

The PROMIS algorithm is expected to improve its performance proportional to the number of the PROMISGROUP members. However, it became evident that we were unable to find enough volunteers that would permit us to install the PROMIS application on their systems. The NetBiotic prototype, which is based on the JXTA framework and is written completely in Java, allows us to install it on various different applications and operating systems. On the other hand, in that way we would be forced to write and support many different Notifier daemons in order to read the security log files from numerous vendors. The Handler would have also to be modified to activate different countermeasures according to the host OS and the installed applications.

To address these shortcomings in our second approach we decided to focus on the prevailing platform for most home users which is the Microsoft Windows with the Internet Connection Firewall as the default security application. Following this approach we were able to support with a single release of the MSPROMIS application all our potential users. It turned out however that the deployment of the MSP2P API had some quite restrictive requirements (IPv6, Teredo). However, the most important drawback was due to the homogeneity of the population that

we used to deploy PROMIS system. In particular most of the systems where suffering from the same vulnerabilities thus they where either susceptible or immune to a specific attack. The main advantage of the PROMIS architecture is that it utilizes the collective knowledge of cooperating peers to estimate the general threat level of the Internet. As different species of malware target different vulnerabilities, systems that are immune to a specific vulnerability are capable of monitoring the escalation of the malicious activity and hence of informing their peers so as to increase their security level to avoid a subsequent infection. Of course no guarantees can be given that a peer even with increased security settings will remain uninfected, but the chances are substantially increased. Without the OS and applications diversity among the PROMISGROUP members, no significant information exchange can take place, which in turn hinders the performance of the MSPROMIS system. Thus, the ease of development and maintenance of one single release of the MSPROMIS tool comes at the cost of having a limited view of the overall malicious activity.

Another important issue that should be taken in to account is that malware epidemics do not happen on a regular or predictable basis. Therefore, it would not be a very practical to wait for a worm epidemic in order to reach scientific conclusions about the efficiency of the PROMIS algorithm. Even then, most of our efforts should and will be concentrated on the defense of the most important digital assets instead of performing scientific experiments during the peak of the crisis.

The above reasons made clear that in order to evaluate the performance of the PROMIS algorithm other means should be considered. We found that the most appropriate methodology is to simulate the operation of the PROMIS system. PROMIS is a distributed cooperative containment system and therefore is affected by many distinct factors. Simulation is the only viable way to isolate and interact with these factors in order to get some indications of the overall systems' performance.

We performed various simulations to measure the effectiveness of the PROMIS algorithm. We separated our experiments into two major categories. In the first one we modified all the parameters that are not related with the PROMIS algorithm, so as to investigate its ability to hinder the propagation of rapid malcode and in the second category we measured the implications of various PROMIS related parameters. We found this distinction important as it can be used to identify whether the PROMIS algorithm is capable of containing the spread of aggressive worms in a first place, and secondly what parts of the PROMIS system have to be fine-tuned in order to maximize its performance and coverage so as to protect against even more aggressive types of malcode. The experiments we performed helped us to discover the most important factors that affect its effectiveness. In particular we found that:

- The thresholds to increase and decrease the security settings of the PROMIS system vastly influence the total performance of the PROMIS system. As one might expect, the lower the activation thresholds are, the better protected the system is, albeit at the cost of being in the restricted secure mode most of the time. Our experiments showed that with aggressive security settings a PROMIS system can protect up to 95% of the PROMISGROUP members.
- The total size of the PROMISGROUP is proportional to the number of the survived nodes. In other words the more nodes participating in the PROMISGROUP the more the chances of avoiding an infection.
- The number of the PROMISGROUP members that during each time interval a given node contacts to estimate the global malicious activity and the the minimum number of incoming rates that is sufficient for a global malicious activity rate estimation are highly dependent on the underlying network topology. To avoid inducing significant overhead to the network, each node randomly contacts a fixed number of other nodes in order to exchange information regarding the malicious activity. On the other hand, we found that although a larger number of contacts per node results in a better network coverage, it takes significant more time for the average malicious activity to reach the threshold level so as to activate the countermeasures. The obvious solution to that problem would be to lower the activation threshold. Subsequent to further reflection, we found that this method could be exploited by a single malevolent peer, which can fraudulently transmit to rest of the nodes very large malicious activity measurements. That could significantly raise the average general malicious activity and consequently force the rest of the nodes to increase their security level without a good reason. Therefore in our design, before the activation of the countermeasures, a node should receive similar information from a minimum number of other nodes, which in essence is a percentage of the nodes that contact during each interval. In homogenous graphs because every node is in direct contact with every other node and all of them share the same view about the ongoing malicious activity thus the effects of this factor are not noticeable. The situation changes completely when we used scale-free or random graphs, in which various trade-offs between these two parameters should take place.
- The history window of the past attack rates that each node logs to calculate the local malicious activity rate doesn't play an important role according to our experiments which is rather counterintuitive at first, but can be explained as in this study we focused exclusively on ultra virulent worms.

Our results indicate that each peer of the PROMISGROUP observes a significant increase of the local malicious activity rate during the early phases of the epidemic (Figure 3). This information is dissiminated via the PROMIS

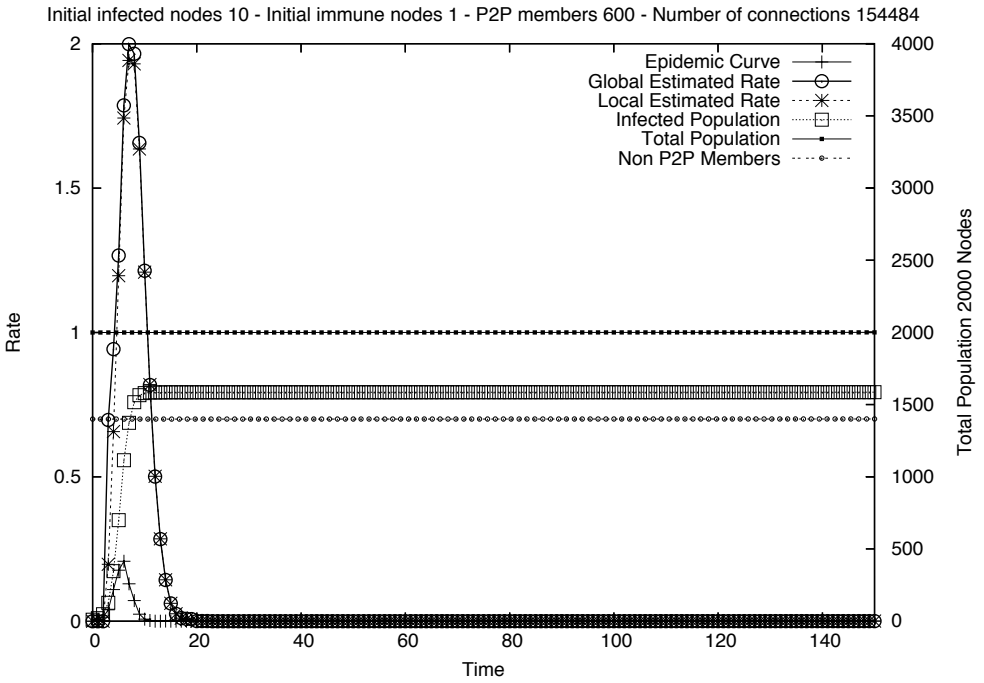

Figure 3: PROMIS simulator output

system to each member of the PROMISGROUP, which in turn calculates the estimated global malicious activity rate of the peer group. Each peer has a different view of the malicious activity as it sustains a different number of attacks, communicates with a divergent set of nodes and therefore calculates its own local and global malicious activity rate. Therefore, in a non-homogeneous graph, such as the scale-free graph, which we used to perform our simulations, each peer has its own perspective of the local and the global malicious activity which might differ slightly from those of other nodes. The curves in Figure 3 depict the average local malicious activity of all the uninfected nodes and the global malicious activity rate of all active PROMISGROUP nodes respectively. The two horizontal lines indicate the total population and the number of the PROMISGROUP members. As one can see from the graph, only a small percentage of the PROMISGROUP members has actually been infected. Figure 4 from another experiment shows the transition of the security levels of the PROMISGROUP nodes that remain uninfected during the course of the epidemic.

As in all complex systems there are various trade offs that are related to the desired security coverage and the available resources. Most importantly parameters that affect the performance of the system heavily are the thresholds which enable the activation of the countermeasures, the total size of the PROMISGROUP, and the minimum number of incoming rates that is sufficient for a global malicious activity rate estimation. Minor impact has the the size of the history window of the past attack rates. On the other hand as we expected the propagation of a virus is highly dependent on the the number of the initial infected nodes, the topology of the network and the pairwise rate of infection.

Our experiments showed that even against unknown worms PROMIS is capable of protecting a significant number of the participating systems. The actual percentage cannot be estimated in advance because many factors affect it, but it can scale up to 95% if we choose to enforce an aggressive security policy with low activation thresholds.

6 Future Work and Concluding Remarks

The simulations we performed in conjunction with the two early prototypes produced positive results. We are planning to build a stable PROMIS system using the experience gained from our past attempts and the experimental results of the simulation. We have decided to base the PROMIS application on the JXTA framework as it is much

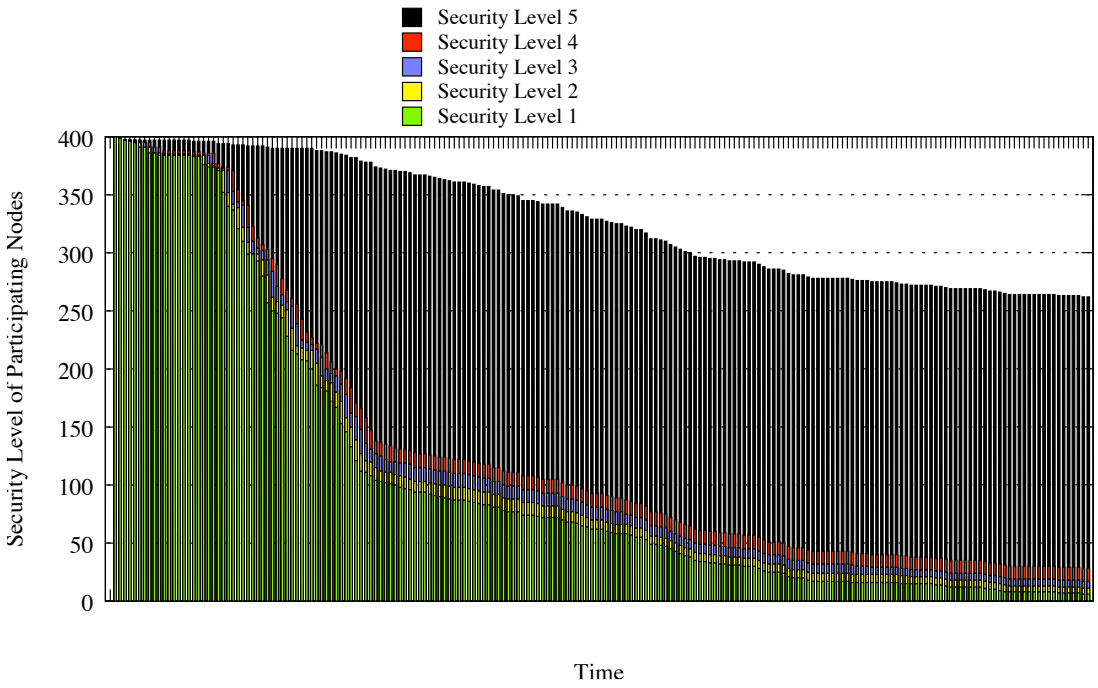

Figure 4: PROMIS peers security level during an epidemic

more mature and widely adopted in comparison to the MSP2P API. Special care should be given to the trust relationships between the PROMISGROUP peer members. We are currently working on a suitable reputation mechanism in order to minimize the possibility of participation of malevolent peers in the PROMIS (Androutsellis-Theotokis *et al.*, 2006).

We presented two proof of concept prototypes that show the technical feasibility of the proposed PROMIS system. Furthermore, extensive simulations suggest that PROMIS is capable, in most cases, of protecting a significant number of systems against recent or unknown worm activity. The benefits of the PROMIS systems are derived from the diversity of operating systems, applications and security mechanisms that are currently in use. It also utilizes the fact that during each malware epidemic only specific vulnerabilities are exploited leaving all the other systems intact. Therefore, by utilizing all the available information from non vulnerable or well hardened systems that have sustained an attack, we are able to alert the susceptible systems and give them more chances of surviving the epidemic. Following the hygiene principles, we try to protect the susceptible systems by automatically increasing their security level as long as the epidemic rages out of control.

Acknowledgments

The work of Vasileios Vlachos is funded under the Iraklitos fellowships for research of Athens University of Economics and Business program and the European Social Fund. We would like to thank Vassiliki Vouzi, Nikos Korfiatis, Andreas Raptis and Martin Papadatos for their invaluable assistance in the prototype development phase. We would like also to thank Anne Leventeris, Vassiliki Tagalaki and Stephanos Androutsellis-Theotokis for their assistance in the compilation of this paper. This paper is an extended and revised version of the "PROMISing Steps Towards Computer Hygiene" paper which appeared in the INC2006 conference (Vlachos *et al.*, 2006). Therefore we were able to incorporate revisions based upon the constructive comments received at the conference and to present the subsequent developments of our work.

References

Adamic, L., & Huberman, B. 2000. Power-Law Distribution of the World Wide Web. *Science*, **287**(2115a).

Anagnostakis, K., Greenwald, M., Ioannidis, S., & Keromytis, A. 2007. Robust Reactions to Potential Day-Zero Worms through Cooperation and Validation. *To appear in the Springer International Journal of Information Security (IJIS), ISC'06 Special Issue*.

Andel, T., & Yasinsac, A. 2006. On the Credibility of Manet Simulations. *IEEE Computer*, **39**(7), 48–54.

Androutsellis-Theotokis, S., & Spinellis, D. 2004. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys*, **36**(4), 335371.

Androutsellis-Theotokis, S., Spinellis, D., & Vlachos, V. 2006 (September). The MoR-Trust distributed trust management system: Design and simulation results. *Page 314 of:* Etalle, Sandro, Foresti, Sara, & Samarati, Pierangela (eds), *Proceedings of the 2nd International Workshop on Security and Trust Management (STM'06), Electronic Notes in Theoretical Computer Science.*

Bailey, M., Cooke, E., Jahanian, F., Watson, D., & Nazario, J. 2005. The Blaster Worm: Then and Now. *IEEE Security & Privacy*, **3**(4), 26–31.

Barabási, A., & Bonabeau, E. 2003. Scale-Free Networks. *Scientific American*, May, 60–69.

Barabási, A., Albert, R., & Jeong, H. 1999. Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A*, **281**, 69–77.

Batagelj, V., & Mrvar, A. 2004 (July). *Program for Analysis and Visualization of Large Networks*. Ljubljana, Slovenia.

Berk, V., Bakos, G., & Morris, R. 2003 (March). Designing a Framework for Active Worm Detection on Global Networks. *In: Proceedings of the IEEE International Workshop on Information Assurance*.

Berners-Lee, T., Fielding, R., Irvine, U., & Masinter, L. 1998. *RFC2396 Current on-line (February 2007): http://www.ietf.org/rfc/rfc2396.txt*. http://www.ietf.org/rfc/rfc2396.txt.

Bornholdt, S., & Ebel, H. 2001. World Wide Web scaling exponent from Simon's 1955 model. *Physical Review E*, **64**(August), 0345104 (R)–1 0345104–4.

Briesemeister, L., & Porras, P. 2005 (June). Microscopic Simulation of a Group Defense Strategy. *Pages 254–261 of: Proceedings of Workshop on Principles of Advanced and Distributed Simulation (PADS)*.

Briesemeister, L., & Porras, P. 2006 (April). Automatically Deducing Propagation Sequences that Circumvent a Collaborative Worm Defense. *Pages 587–592 of: Proceedings of the 25th International Performance Computing and Communications Conference (Workshop on Malware)*.

Chen, Z., Gao, L., & Kwiat, K. 2003 (March). Modeling the Spread of Active Worms. *In: Proceedings of the IEEE Infocom*. San Francisco, USA.

Cohen, R., K.Erez, ben Avraham, D., & Havlin, S. 2000. Resilience of the Internet to Random Breakdowns. *Physical Review Letters*, **85**(21), 4626.

Coull, S., & Szymansky, B. 2005. *A Reputation-based System for the Quarantine of Widespread Malicious Behaviour*. Tech. rept. 05-01. Rensselaer Polytechnic Institute.

Daley, D., & Gani, J. 1999. *Epidemic Modelling*. Cambridge, UK: Cambridge University Press.

DeepSight. 2007. *Symantec DeepSight Threat Management System Current on-line (November 2007): http://tms.symantec.com/.*

DShield. 2007. *Distributed Intrusion Detection System. :Current on line (December 2007): http://www.dshield.org/.*

Ebel, H., Mielsch, L., & Bornloldt, S. 2002. Scale-free topology of e-mail networks. *Physical Review*, **E 66**(035103(R)).

Faloutsos, M., Faloutsos, P., & Faloutsos, C. 1999. On Power-Law Relationships of the Internet Topology. *Pages 251–262 of: Proceedings of ACM SIGCOMM*.

Goldenberg, J., Shavitt, Y., Shir, E., & Solomon, S. 2005. Distributive immunization of networks against viruses using the 'honey-pot' architecture. *Nature Physics*, **1**, 184–188.

Janakiraman, R., Waldvogel, M., & Zhang, Q. 2003 (June). Indra: A peer-to-peer aroach to network intrusion detection and prevention. *In: Proceedings of 2003 IEEE WET ICE Workshop on Enterprise Security.*

Kanovsky, I., & Mazor, S. 2003 (July). Models of Web-Like Graphs: Integrated Approach. *Pages 278–283 of: Proceedings of the 7th World Multiconference on Systematics, Cybernetics and Informatics (SCI 2003)*.

Kephart, J. 1992 (June). How topology affects population dynamics. *In: Proceedings of Artificial Life 3*.

Kephart, J., & White, S. 1999 (May). Measuring and Modeling Computer Virus Prevalence. *Pages 2–14 of: Proceedings of the 1999 IEEE Computer Society Symposium on Research in Security and Privacy*.

Kephart, J., Chess, D., & White, S. 1993. Computers and Epidemiology. *IEEE Spectrum*, **30**(20).

Kim, H., & Kang, I. 2004 (June). On the functional validity of the worm-killing worm. *Pages 1902– 1906 of: Proceedings of the 2004 IEEE International Conference on Communications*, vol. 4.

Law, A., & Kelton, W. 2000. *Simulation Modeling and Analysis*. Third edn. McGraw-Hill.

Leveille, J. 2002 (October). *Epidemic Spreading in Technological Networks*. HPL-2002-287, School of Cognitive and Computing Sciences, University of Sussex at Brighton, Bristol.

Medina, A., Matta, I., & Byers, J. 2000. On the Origin of Power Laws in Internet Topologies. *ACM Computer Communication Review*, **30**(2), 160–163.

Microsoft. 2003a. *Teredo Overview: Current on-line (November 2006): http://www.microsoft.com/technet/prodtechnol/winxppro/maintain/Teredo.mspx*.

Microsoft. 2003b. *Windows Peer-to-Peer Networking: Current on-line (November 2006): http://www.microsoft.com/technet/itsolutions/network/p2p/default.mspx*.

Microsoft. 2006. *Introduction to IP Version 6*. Tech. rept. Microsoft Corporation.

Middleton, J. 2001 (September). *Anti-worms' fight off Code Red threat,*. http://www.vnunet.com/News/1125206. Current on-line (November 2005):.

Moore, D., Paxson, V., Savage, S., Shannon, C., Staniford, S., & Weaver, N. 2003a. Inside the Slammer Worm. *IEEE Security & Privacy*, July 2003, 33–39.

Moore, D., Shannon, C., Voelker, G., & Savage, S. 2003b (April). Internet quarantine:Requirements for containing self-propagating code. *In: Proceedings of 22nd Annual Joint Conference of IEEE Computer and Communication Societies (INFOCOM 2003)*.

Moore, David, Shannon, Colleen, & Brown, Jeffery. 2002. Code-Red: a case study on the spread and victims of an Internet worm. *In: Proceedings of the Internet Measurement Workshop*.

Palmer, C., & Steffan, J. 2000 (December). Generating Networks Topologies That Obey Power Laws. *In: Proceedings of the GLOBECOM 2003*.

Pastor-Satorras, R., & Vespignani, A. 2001. Epidemic spreading in scale-free networks. *Physical Review Letters*, **86**, 3200–3203.

Paxson, V., & Floyd, S. 1997 (December). Why we don't know how to simulate the internet. *In: Proceedings of the Winder Communication Conference*.

Scandariato, R., & Knight, J. 2004 (October). *An Automated Defense System to Counter Internet Worms*. Submitted to SRPS 2004, 23rd Symposium on Reliable Distributed Systems. Florianapolis, Brazil.

Schneier, B. 2006. University Networks and Data Security. *IEEE Security & Privacy*, **4**(5), 88.

Shannon, C., & Moore, D. 2004. The Spread of the Witty Worm. *IEEE Security & Privacy*, **2**(4), 46–50.

Spinellis, D. 2000 (June). Outwit: Unix Tool-Based Programming Meets the Windows World. *Pages 149–158 of: Proceedings of the USENIX 2000 Technical Conference*.

Staniford, S. 2003. Containment of Scanning Worms in Enterprise Networks. *Journal of Computer Security*, October.

Staniford, S., Paxson, V., & Weaver, N. 2002 (August). How to 0wn the Internet in Your Spare Time. *Pages 149–167 of: Proceedings of the 11th USENIX Security Symposium*.

Staniford, Stuart, Moore, David, Paxson, Vern, & Weaver, Nicholas. 2004. The top speed of flash worms. *Pages 33–42 of: WORM '04: Proceedings of the 2004 ACM workshop on Rapid malcode*. New York, NY, USA: ACM Press.

Tangmunarunkit, H., Govindan, R., Jamin, S., Shenker, S., & Willinger, W. 2002 (August). Network Topology Generators: Degree-Based vs. Structural. *Pages 147–159 of: Proceedings of ACM SIGCOMM '02*.

Vlachos, V., Androutsellis-Theotokis, S., & Spinellis, D. 2004. Security applications of peer-to-peer networks. *Comput. Networks*, **45**(2), 195–205.

Vlachos, V., Vouzi, V., Chatziantoniou, D., & Spinellis., D. 2005a. NGCE Network Graphs for Computer Epidemiologists. *Pages 672–683 of:* Bozanis, Panagiotis, & Houstis, Elias N. (eds), *In Advances in Informatics: 10th Panhellenic Conference on Informatics, PCI 2005, Lecture Notes in Computer Science 3746*. Springer-Verlag.

Vlachos, V., Vouzi, V., & Spinellis, D. 2005b (May). PROMIS: PROactive Malware Identification System. *In: Department of Management Science & Technology (DMST) 2nd Conference (in Greek)*.

Vlachos, V., Raptis, A., & Spinellis, D. 2006 (July). PROMISing Steps Towards Computer Hygiene. *Pages 229–236 of:* Furnel, Steven (ed), *International Network Conference (INC2006)*.

Wang, C., Knight, J., & Elder, M. 2000 (December 11-15). On Computer Viral Infection and the Effect of Immunization. *Pages 246–256 of: Proceedings of the 16th Annual Computer Security Applications Conference (ACSAC),New Orleans, Louisiana, USA*.

Weaver, N., Paxson, V., & Staniford, S. 2004 (May). A Worst-Case Worm. *In: Proceedings of the Third Annual Workshop on Economics and Information Security (WEIS04)*.

Zou, C., Gong, W., & Towsley, D. 2002 (November). Code Red Worm Propagation Modeling and Analysis. *In: Proceedings of the 9th ACM Conference on Computer and Communication Security (CCS)*.

Zou, C., Towsley, D., & Gong, W. 2003a (May). *Email Virus Propagation Modeling and Analysis*. Tech. rept. Umass ECE TR-03-CSE-04.

Zou, C., Gao, L., Gong, W., & Towsley, D. 2003b (October). Monitoring and Early Warning for Internet Worms. *In: Proceedings of the 10th ACM Conference on Computer and Communication Security*.