*Proceedings of the 2010 Winter Simulation Conference*
*B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, eds.*

# CYBERSIM: GEOGRAPHIC, TEMPORAL, AND ORGANIZATIONAL DYNAMICS OF MALWARE PROPAGATION

Nandakishore Santhi
Guanhua Yan
Stephan Eidenbenz

Computer and Computational Sciences
Los Alamos National Laboratory
Mail Stop: B256, Los Alamos, NM 87545, USA

## ABSTRACT

Cyber-infractions into a nation's strategic security envelope pose a constant and daunting challenge. We present the modular CyberSim tool which has been developed in response to the need to realistically simulate at a national level, software vulnerabilities and resulting malware propagation in online social networks. CyberSim suite (a) can generate realistic scale-free networks from a database of geo-coordinated computers to closely model social networks arising from personal and business email contacts and online communities; (b) maintains for each host a list of installed software, along with the latest published vulnerabilities; (c) allows to designate initial nodes where malware gets introduced; (d) simulates using distributed discrete event-driven technology, the spread of malware exploiting a specific vulnerability, with packet delay and user online behavior models; (e) provides a graphical visualization of spread of infection, its severity, businesses affected etc to the analyst. We present sample simulations on a national level network with millions of computers.

## 1 INTRODUCTION

The combined real cost incurred by individuals, small businesses and large industries as a result of malware infecting otherwise productive computer networks is often hard to estimate. The actual loss in productivity, perceived lack in consumer confidence due to explicit loss of privacy, irreplaceable loss of data, loss in hardware efficiency because of the malware running in the background, costs associated with installing and maintaining anti-malware systems are all prominent factors.

The actual extent and nature of malware infections on a large scale computer network is decided by the specific software vulnerabilities which are exploited by a particular piece of malicious software. A malware spreads over a network by making effective use of the small diameter properties of social networks induced by email-contacts and various online communities: it is therefore crucial to include the specific nature of the social network in any serious study.

When estimating the economic impact of an exploit affecting a popular piece of software, or a popular social network, it is necessary to consider relatively large and complex computer networks with hundreds of millions of hardware hosts. Each host or device is in turn associated with a myriad of software applications which are installed on them and are routinely accessed by individual users. Further, an accurate model for the flow of packets in Internet settings is also called for. Any simulation tool attempting to address this challenge will have to be designed in a scalable, distributed manner (Floyd and Kohler 2002; Fujimoto et al. 2003).

The rate at which a malware propagates over a network is further determined by the frequency and duration of online presence of a network user as well as the usage rate of a particular vulnerable piece of software application installed on his or her computer.

Vulnerabilities and exploits of commonly used software applications are compiled by computer security interest groups and organizations. A reliable source of such information exists in the form of a Common Vulnerabilities and Exposures (CVE) database maintained by NIST: the National Vulnerabilities Database (NVD) (NIST 2010). There are some open source and commercial software specializing in analyzing and estimating the impact of these vulnerabilities. Most of these are listed online (MITRE 2009; SecurityFocus 2010). NIST itself provides a simple online form based CVE analysis facility.

The CyberSim suite was designed at the Los Alamos National Laboratory (LANL) to address all these modeling and simulation challenges, while at the same time being flexible and scalable. The CyberSim tool is intended to be easy to use, providing an intuitive visualization interface. The CyberSim toolkit includes a network generation module, which takes as input various parameters which are widely accepted to be representative of a social network model: scale-free nature induced through a fixed power-law degree distribution, and clustering based on geographic proximity of social contacts. The network generation module generates a social network according to the model parameters from a database of Internet hosts with geo-location information gathered from Internet service provider surveys. Traditional tools for malware analysis typically lack such an ability to model social networks in a realistic manner. The social network parameters taken as input by the network generation module are learned from real world networks such as email contact lists and online community graphs as a result of surveys. CyberSim has the ability to model various types of social networks: personal email contact lists, business contact lists and online communities. The toolkit has the ability to capture the inter-connections between several types of social networks at the same time.

CyberSim includes a highly distributed, scalable discrete event network simulation module built upon the SimCore framework (Kroc, Eidenbenz, and Ramaswamy 2007a) developed at LANL. This module allows the simulation of social networks with millions of networked devices on heterogeneous distributed platforms consisting of a grid of multi-core computers, each with limited physical memory. SimCore in turn relies on a simulation engine such as PrimeSSF (PRIME 2010) and MPICH2 (MPICH2 2010). SimCore has been previously used in building complex simulation tools such as Multiscale Integrated Information and Telecommunications System (MIITS) (Waupotitsch et al. 2006; Ramaswamy et al. 2007), SessionSim (Kroc, Eidenbenz, and Ramaswamy 2007b), ActivitySim (Galli et al. 2009), FastTrans (Thulasidasan et al. 2010), and BotSim (Ha et al. 2009).

In this paper, in Section 2 we introduce the simulation flow and various modules within the CyberSim suite in detail, then in Section 3 we present some simulation studies on a large scale realistic social network. In Section 2.2 and 2.3 we introduce the Social Network Generation module, along with the methodology adopted in the populating host devices. In section 2.4 and 2.5 we introduce the Network Simulation module and the SimCore simulation interface. In Section 2.6 we introduce the Processing and Visualization module. In Section 3, we present a few results from a study conducted on a large scale social network.

## 2 CYBERSIM OVERVIEW

The main design objective of CyberSim is to estimate the impact of software vulnerabilities on various business sectors, its severity as well as rapidity. Good estimates of the dynamics of malware spread can in turn be used to come up with effective remedial actions. The availability of a scalable, flexible and realistic analysis tool provides engineering guidance in placement of packet filters and firewalls to prevent rapid and catastrophic spread of malware on critical networks.

Figure 1 shows the layered architecture of CyberSim. The Foundation Framework is the core layer providing the core support libraries (such as SimCore) for the other layers and modules. The Host Device Generation module populates a database of Internet enabled hardware hosts from data gathered through surveys. The Software Registry Generation module populates each hardware device with a software registry of installed operating systems, and Internet applications such as media players, browsers, email-clients and office documentation software.

The Vulnerabilities Database Manager module is responsible for periodically updating the database of known vulnerabilities for the installed software on the hardware nodes. The Social Network Modeling module takes as input the databases maintained by the Host Device Generation, Software Registry Generation and Vulnerabilities Manager modules in order to generate a realistic social network. The Network Simulation module uses a distributed message passing simulator to simulate the spread of a malware over the social network model. The Processing and Visualization module generates detailed true-scale time lapse geo-coordinated information on the malware spread and its impact on various businesses in a commonly supported markup language such as KML.

The Social Network Modeling module of CyberSim is capable of modeling a wide variety of social networks by capturing the salient features of realistic social networks such as their scale free nature, and geographical proximity based clustering. Figure 2 illustrates the various types of social networks that CyberSim is currently capable of modeling.

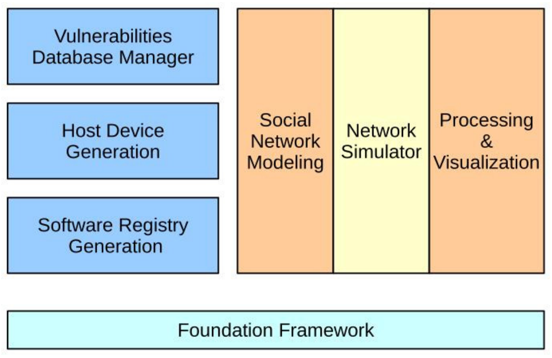
Vulnerabilities Database Manager
Host Device Generation
Software Registry Generation
Social Network Modeling
Network Simulator
Processing & Visualization
Foundation Framework

Figure 1: CyberSim architecture

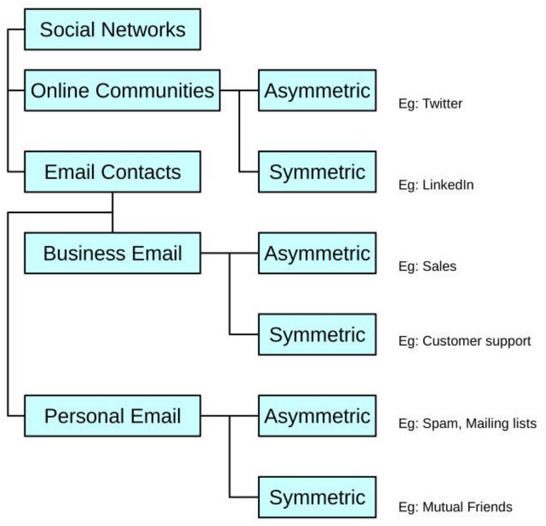

Figure 2: Types of social networks which can be modeled using CyberSim

## 2.1 CyberSim Simulation Flow

CyberSim has been designed to be extensible and flexible, while being powerful enough to simulate with ease social networks consisting of millions of host devices and users. Hence, the system had to be engineered to run in parallel on thousands of computing units. We chose the SimCore library which has been developed and proven at LANL over the last decade to provide the core foundation for the CyberSim network simulation modules, thus isolating the distributed simulation system design details from the specific task at hand.

As illustrated in Figure 1, the CyberSim Network Simulation module takes as input the social network generated by the Social Network module. The Social Network Modeling module in turn relies on data produced by the Host Device and Software Registry Generator modules. The Network Simulator is further capable of taking as input the online activity of individual users, and the initial list of nodes where a malware originates. The Network Simulator module includes a simple IP layer simulator which realistically models the packet propagation over the Internet. The Network Simulator module outputs a list of entities infected in the course of a malware propagation, which is processed for visualization in a markup language (KML) format by the Processing and Visualization module.

## 2.2 Social Network Modeling Module

The Network Modeling Module takes as input two types of profiles: (a) the degree distribution profile - it could be a power-law distribution to induce a scale-free behavior, or an empirical histogram learned from a real life social network; and (b) the distance distribution profile for the social contacts (neighbors in the social graph) – this enforces a sort of local clustering and sub-community structure. The other input to the Modeling module is a database of hardware hosts (computers) with a software registry associated with each host.

In order to generate a typical social network satisfying the given degree and distance profiles, for every device-node *i* in the graph, the degree profile is first sampled to get a degree $d_i$ for the node *i*. Now for each index *j* in *{1, ..., $d_i$}*, a distance $D_j$ is sampled from the distance profile. Now a device $n_j$ at distance $D_j$ relative to *i* from the device list is added as a neighboring node to the given node *i* along with the edge *(i,$n_j$)*. Figure 3 illustrates a typical social network generated in this fashion.

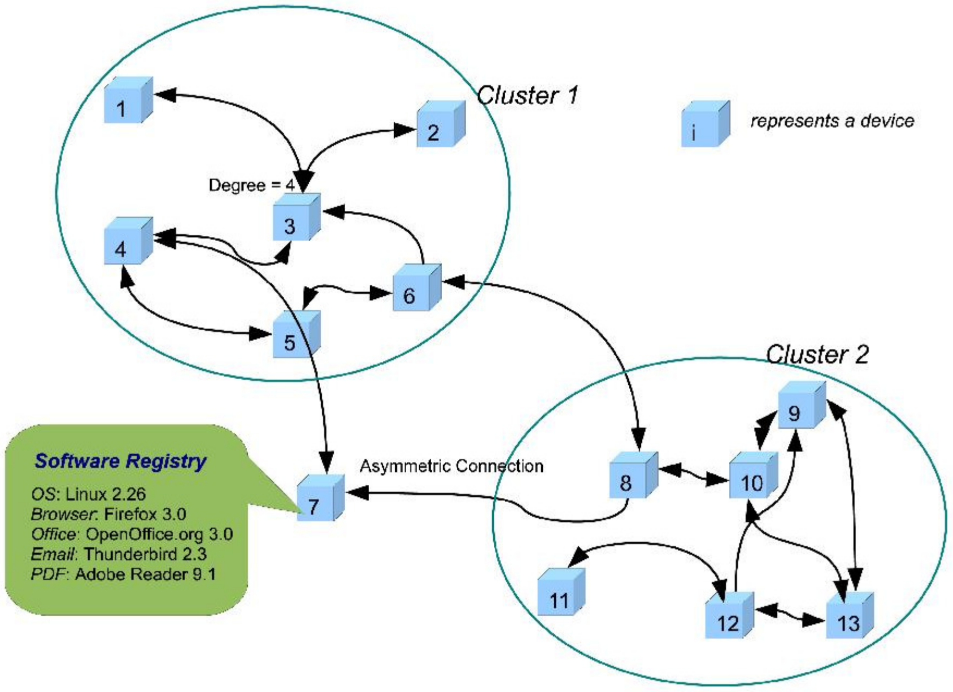

Figure 3: A typical social network is generated to match a given degree and distance distribution. The hosts are then populated with software registries.

This social network modeling approach is extensible in a variety of ways. For example, for modeling social contacts representing business relations, we can first categorize the network nodes according to business sector and organizational function. Then the contacts are sampled from the appropriate categories according to their relative probability distribution.

## 2.3 Software Registry Generation Module

Each network host has a set of installed software applications which are potentially vulnerable. In order to populate what we call the *software registry* on each host, the registry generation module is used. We gathered market share data (StatOwl 2009; FingerPrintApp 2009) on popular software applications. Software categories which we consider include operating systems, web browsers, email clients, document viewers, multimedia player, office documentation suites and plug-ins such as flash and silverlight. The

Software Registry Generation module samples the market share statistics to populate each host in a network with a registry of installed software. This software registry is used as an input to the network simulation module and has entries with a format given below:

```
host-id [Home|Business] CategoryName:SoftwareName+VersionName  Ca-
tegoryName:SoftwareName+VersionName|SoftwareName+VersionName
```

## 2.4 Network Simulation Module

Figure 4 depicts the flow chart of simulating malware spreading in social networks. Each host object consists of three components: Software Registry, Social Network Client, and Worm Propagation. The Software Registry component maintains the list of software installed on a host machine. In the simulation environment, we model the installation and dis-installation of software applications explicitly as events and such a treatment enables us to model human intervention as a response to malware attacks in a dynamic environment. The Social Network Client component models the behaviors of Internet users when dealing with a variety of communication messages in the cyber space, such as instant messages and Emails. The Worm Propagation component deals with malware spreading using online contacts maintained by the Social Network Client Component.

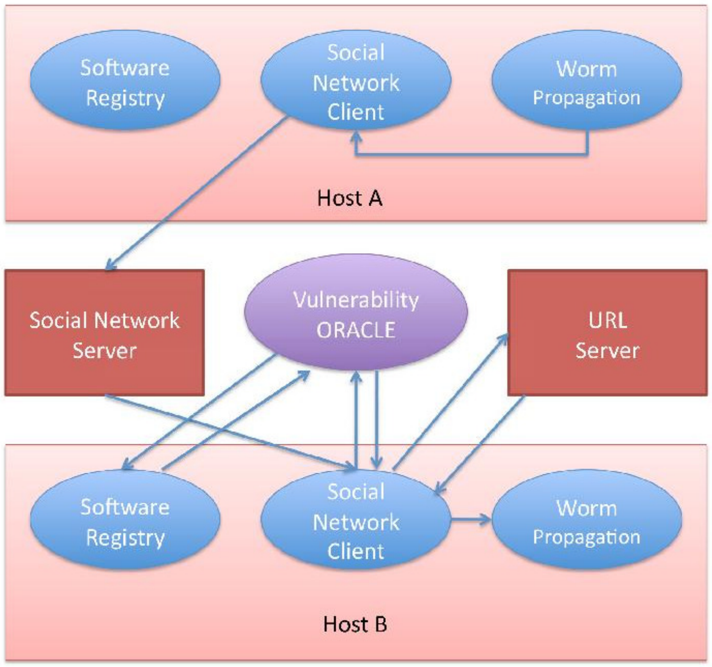

Figure 4: Network simulation module flowchart showing object interactions.

In addition to the modules belonging to each host machine, we also model two types of servers in CyberSim explicitly. The Social Network Server component is responsible for processing communication messages sent from the Social Network Client components and delivering them to the destined host machines accordingly. A URL server models a typical HTTP server, which accepts requests for file downloading. In our model, a URL server can be used for hosting malware programs. Upon the receipt of a file download request, such a server sends the malware code to the requesting machine. The Network Simulation module includes the Vulnerability Oracle component, which models the logic (residing in the malware code) of whether a machine is susceptible given a particular CVE id and the list of software installed on it.

We now discuss how CyberSim models malware spreading over social networks in the cyber space. Suppose that Host A has been infected by the malware. The malware instance on Host A obtains a list of contacts from the Social Network Client component (which abstractly models a typical malware behavior) on the same machine, and sends a message carrying a malicious URL to each of its contacts. All these messages are delivered to the corresponding recipients through the Social Network Server. Assume that Host B is one of the machines that receive a malicious message from Host A. When the Social Network Client component on Host B receives the message, it downloads the malware program from the encoded URL. It further requests the Vulnerability Oracle to check whether Host B is vulnerable to the exploit in the malware code. The Vulnerability Oracle checks the software types installed on Host B and their version numbers and patch levels. Then using the list of vulnerabilities collected from the NVD, decides whether Host B is susceptible to infection. Once the Social Network Client component receives the decision from the oracle and finds that Host B is vulnerable but not infected yet, it initiates another infection cycle. The process repeats until no more hosts can be infected.

## 2.5 SimCore Discrete Event Simulation Framework

The Foundation Framework consists of the SimCore and supporting libraries such as PrimeSSF (PRIME 2010). SimCore was designed at LANL to address the need for an *entity-service* level interface on which application specific simulation systems could be built. Figure 5 shows how SimCore facilitates integration of CyberSim's network simulation module with PrimeSSF.

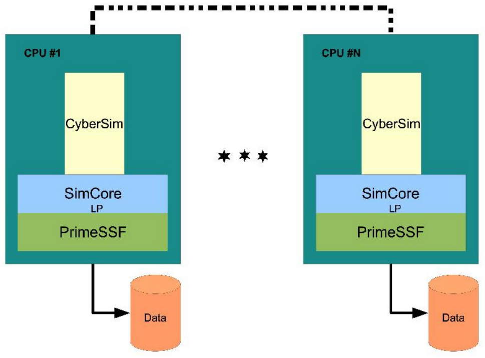

Figure 6: SimCore provides APIs to develop simulation systems using PrimeSSF as its underlying simulation engine. The main function of SimCore is to isolate the details of the distributed simulation system (such as logical processes and CPUs) from CyberSim's network simulation module.

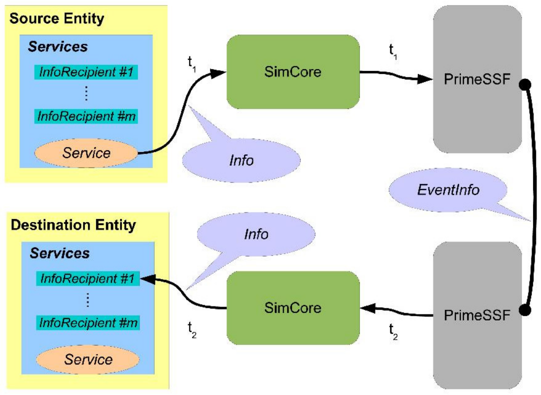

Figure 5: Flow of an Info Package from a Source Entity to a Destination Entity using SimCore.

SimCore isolates the implementation details of the distributed simulation engine (such as PrimeSSF) from the end system design. The end system designer designs the system in terms of entities with services which handle packets which are sent from other entities. The data input to the simulation system is managed through any combination of entities and services. It is not necessary for the end-system designer to know about the actual computational node and logical process where each entity resides – this is facilitated by SimCore if the designer follows the API specifications of SimCore. SimCore in turn relies on a conservatively synchronized discrete event simulation engine such as PrimeSSF. It also provides a comprehensive set of event logging and input/output facilities. Figure 6 shows how an Info package goes through SimCore mediation, from a source entity at time $t_1$ in some CPU or LP to a recipient service in a destination entity (which could reside in another CPU/LP) to be received at time $t_2$.

### 2.6 Analysis and Visualization Module

This module is responsible for post-processing the output of the Network Simulation module. The module outputs the results in a form which is easily analyzed. One of the functions of the module is to generate a true scale, time-lapse animated data which shows the geographical extent and nature of the spread of malware. The data to be visualized is output in an extensible markup language – we use the KML format, which is supported for example by the *Google Earth* application. Another function of the analysis module is to estimate the impact of the malware on various business sectors.

## 3 A LARGE SCALE SOCIAL NETWORK SIMULATION

In this section we present simulation results using a large scale social network with almost a million networked devices, geographically spread across the United States (Yan et al. 2010). We begin by describing the methodology used in populating the social network, software registries and user sessions.

### 3.1 Methodology Used for Populating the Social Network

Collecting data related to social networks is the most time consuming step in our simulation cycle. We statistically analyzed databases of email contact lists and online communities to generate power-law degree distribution profiles which are a widely accepted social network parameter. We also statistically estimated the distance distribution of neighbors on typical online social networks. The distance distribution profile effectively captures the local clustering of social contacts.

Next we analyzed the asymmetry in several common social networks. By asymmetry, we mean the ratio of mutually non-reciprocated social contacts to the total contacts for a given individual. We then populated a database of Internet connected hardware hosts with their latitude and longitude locations. The data on the hardware devices came from a survey of Internet service providers. Once we gathered and analyzed data in this manner, we generated social network models from the devices database, so as to closely resemble a realistic network. We also gathered market share data on popular software applications. Software categories which we consider include operating systems, web browsers, email clients, document viewers, multimedia players, office documentation suites and plug-ins such as flash and silverlight. We used the market share data to populate each node in the generated social network with a software-registry of installed applications. The Internet users were modeled to be online according to a Poisson process at an average rate of two per day. When the users get online, and use a vulnerable software, they become susceptible to malware which exploits the given vulnerability.

### 3.2 Results

We ran simulations on a generated social network model with about 800,000 Internet enabled hosts geographically distributed throughout the United States. The social network which we simulated was a mix of email and online social networking. Degree and distance distributions of contacts in the online social network were used as the network parameters.

The Common Vulnerabilities and Exposures id *cve-2009-1136* listed in the NVD was chosen for compiling our simulation results. This CVE id refers to the Microsoft Office Web Components HTML Script Vulnerability, which has been exploited in the wild during July to August 2009 (CVE 2009). Microsoft Office Web Components Spreadsheet ActiveX control distributed in various Microsoft Office software versions is considered vulnerable to this CVE. Only computers installed with the susceptible versions are likely to be infected by online malware propagation. This particular vulnerability has been rated as having a severity level of 9.3 (HIGH) (NVD 2009).

### 3.2.1 Degree and Distance Distribution Profiles

The degree distribution and distance distribution profiles were gathered from analysis on real world networks. The degree distribution of Figure 7 determines the average connections that each entity in the network has, while the distance distribution shown in Figure 8 gives the proximity to any of their social contacts. The degree profile shows an approximate power-law behavior, while the distance profile shows a very pronounced preference for local social contacts.

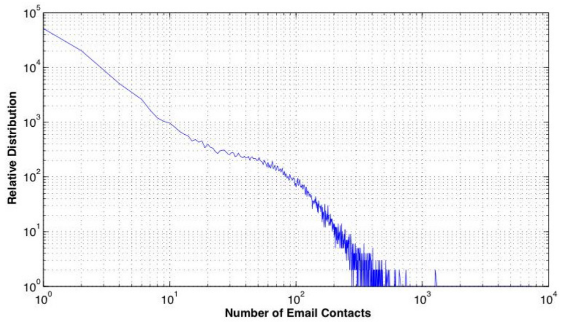

Figure 7: Typical degree distribution of a social network generated by the Social Network Generation module. The network represents a symmetric email contact list.

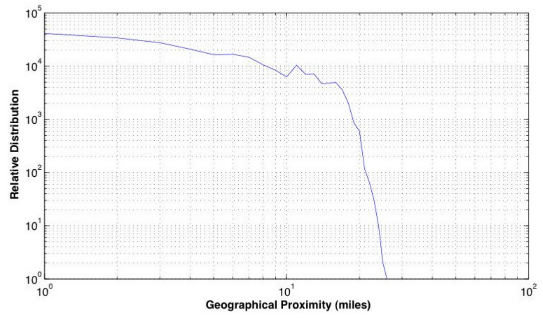

Figure 8: The distance distribution profile of an online community generated by the Social Network Generation module.

### 3.2.2 Spread of Malware as a Function of Time

We simulated the spread of a malware exploiting *cve-2009-1136*. The number of hosts infected with the malware is shown as a function of time in Figure 9. We see that, once a critical number of hosts get infected, the spread of malware is very rapid. The rate of spread depends critically on the connectivity of the initially infected nodes, as well as on the popularity of the vulnerable software version. For the purposes of this paper, we chose a random node (which happens to fall in California) as our initially infected node.

A scatter plot with the distribution of infected hosts as a function of time and distance from the malware originating node is shown in Figure 10. The scatter plot shows the initial stages of malware spread in more detail. Due to the long range nature of some of the social contacts, there is a relatively wide geographical spread early on. As the simulation progresses we can also see pronounced bunching with distance as a result of nodes having more contacts in nearby locales.

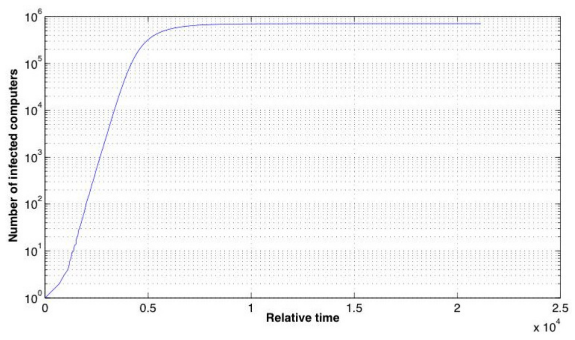

Figure 9: The number of computers infected with the malware as a function of time.

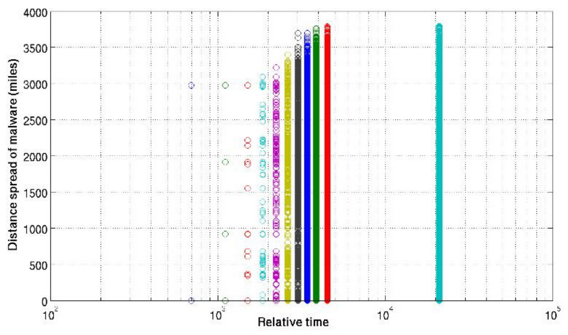

Figure 10: Scatter plot of infected computers in the network as a function of time and distance from the malware origination node.

### 3.2.3 Geographic Visualization of Dynamics of Malware Infection

A snippet of a log output generated by the Network Simulation module is given below:

```
Mar 30 15:56:13.805206 t00     CyberSim  INFO: ============= Sim-
   Time: 2040 =============
Mar 30 15:56:13.805245 t00     CyberSim  INFO: SocialNetworkPropa-
   gation::receive(InfectionAndSpreadInfo) (d 10000974192): received
   info: InfectionAndSpreadInfo with contents= url:999900
Mar 30 15:56:13.805257 t00     CyberSim  INFO: Host (d 10000974192)
   gets infected at time 2040 ...
```

The Processing and Visualization module parses this log and generates a KML animation as below:

```
<Document> <name>CyberSim Output</name> <open>0</open>
   <Style> <IconStyle id="iconstyle"> <scale>1</scale> </IconStyle>
   </Style>
   <Placemark id="d60008790108"> <name>60008790108</name>
   <visibility>1</visibility>
   <description> Time: 100.&lt;br&gt; </description>
   <Point><coordinates>-117.770009,33.530207,0</coordinates></Point>
   </Placemark> ...
```

The extent and rapidity of malware infection is now easily observed when the network can be interactively visualized on a map. We used the Google Earth application to view the KML files generated by the Processing and Visualization module. The spread of malware can be interactively visualized in a true time-scaled movie. The analyst can pause, replay and zoom into areas of interest at any point during the playback. The analyst can adjust the simulation to start from different sets of initially infected hosts and CVE scenarios. Various stages of malware dynamics are apparent in the different frames in Figure 11.

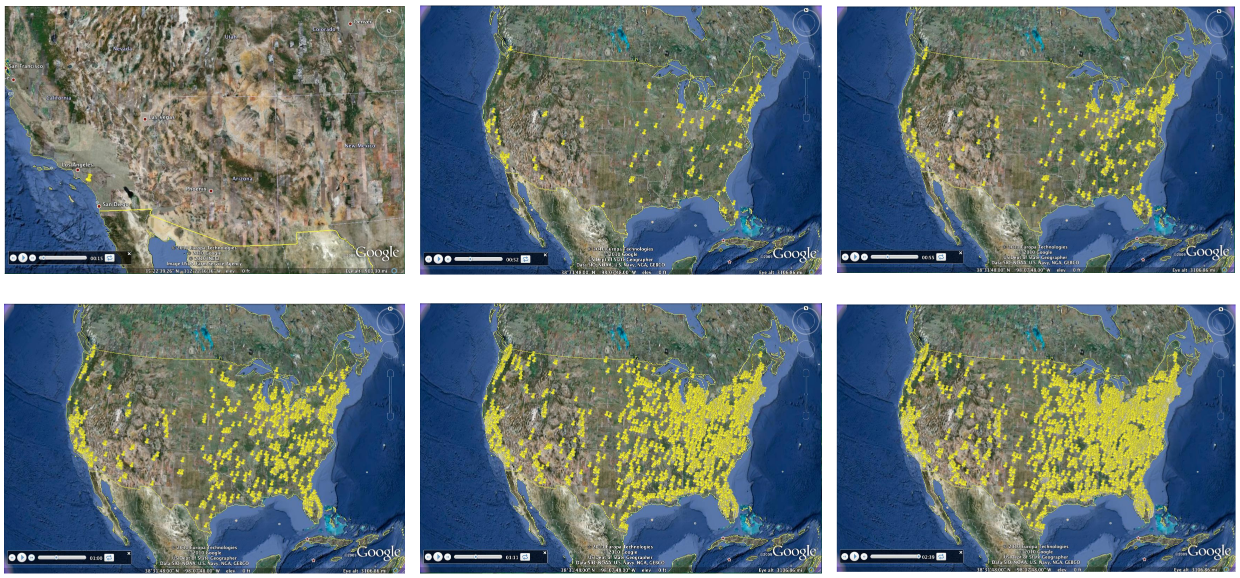
Figure 11: Time lapse view of malware propagation. Several frames of the KML animation as viewed using the *Google Earth* application are shown.

### 3.2.4 Performance

The average memory usage was less than 3300 bytes per social network node, including the memory used to store all associated data structures. In order to compile scaling results, we ran simulations on a Linux cluster with MPICH2, a shared physical memory of 12GB, using up to four 64 bit processors running at 3.33 GHz each. The scaling tests yielded the plot of Figure 12.

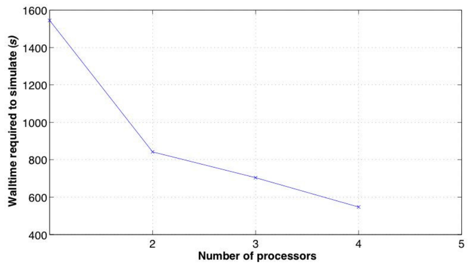

Figure 12: Scaling with the number of CPUs

Increasing the number of processors initially results in a corresponding scaling in the time required for simulation. Subsequently the gain due to parallelization decreases, due to messaging and synchronization overheads. We observed that distributing Social Network Server entities across the LPs helps improve the performance over having a common Social Network Server entity due to reduced messaging.

## 4 CONCLUSION

We presented the CyberSim distributed malware propagation simulation system for large online social networks. CyberSim has an extensible modular architecture. We generated a realistic social network with about a million networked entities and presented the results of simulating a malware propagation which exploits the CVE id *cve-2009-1136*.

An area for future improvement is the way in which CyberSim deals with inter-dependencies among personal and business social networks as well as those due to network hosts being shared among multiple users. There is a need to improve how interactions between different organizational roles are modeled while generating the social network.

## REFERENCES

CVE 2009. Entry in Common Vulnerabilities and Exposures Candidate List, Available via <http://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2009-1136> [accessed June 12, 2010].

FingerPrintApp 2009. Available via <http://fingerprintapp.com/email-client-stats> [accessed August 1, 2009].

Floyd, S., and E. Kohler 2002. Internet Research Needs Better Models, In *Proceedings of the First ACM SIGCOMM Workshop on Hot Topics in Networks* (*Hotnets-I)*, Princeton, New Jersey, USA.

Fujimoto, R. M., K. S. Perumalla, A. Park, H. Wu, M. H. Ammar, and G. F. Riley 2003. Large-Scale Network Simulation: How Big? How Fast? In *Proceedings of the 11th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*.

Galli, E., L. Cuéllar, S. Eidenbenz, M. Ewers, S. Mniszewski, and C. Teuscher 2009. ActivitySim: large-scale agent-based activity generation for infrastructure simulation. In *Proceedings of the 2009 Spring Simulation Multiconference.* San Diego, California, USA.

Ha, D. T., G. Yan, S. Eidenbenz, and H. Q. Ngo 2009. On the effectiveness of structural detection and defense against P2P-based botnets. In *Proceedings of the 39th Annual IEEE/IFIP International Conference on Dependable Systems and Networks,* 297-306.

Kroc, L., S. Eidenbenz, and V. Ramaswamy 2007a. Simcore. *Los Alamos National Laboratory Unclassified Technical Report,* LA-UR: 07-0590.

Kroc, L., S. Eidenbenz, and V. Ramaswamy 2007b. Sessionsim. *Los Alamos National Laboratory Unclassified Technical Report,* LA-UR: 07-0592.

MITRE 2010. CVE Compatible Software List. Available via <http://cve.mitre.org/compatible/compatible.html> [accessed May 18, 2010].

MPICH2 2010. MPICH2: Portable Implementation of Message Passing Interface Standard. Available via <http://www.mcs.anl.gov/research/projects/mpich2/> [accessed May 20, 2010].

NIST 2010. National Vulnerabilities Database maintained by NIST. Available via <htttp://web.nvd.nist.gov> [accessed June 12, 2010].

NVD 2009. National Vulnerabilities Database CVE Search Facility, Available via <http://web.nvd.nist.gov/view/vuln/search> [accessed June 12, 2010].

PRIME 2010. PRIME: Parallel Real-time Immersive network Modeling Environment. Available via <https://www.primessf.net/bin/view/Public> [accessed May 20, 2010].

Ramaswamy, V., S. Thulasidasan, P. Romero, S. Eidenbenz, and L. Cuéllar 2007. Simulating the national telephone network: A Socio-Technical Approach to Assessing Infrastructure Criticality. In *Proceedings of the 2007 IEEE Military Communications Conference,* 1–7.

SecurityFocus 2010. Security Focus Vulnerability Database maintained by Symantec Corporation. Available via <http://www.securityfocus.com/bid> [accessed May 20, 2010].

StatOwl 2009. Available via <http://www.statowl.com/index.php> [accessed August 1, 2009].

Thulasidasan, S., S. Kasivishwanathan, S. Eidenbenz and P. Romero 2010. Spatial Scattering for Load Balancing in Conservatively Synchronized Parallel Discrete-Event Simulations. In *Proceedings of 24th ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation*.

Waupotitsch, R., S. Eidenbenz, J. P. Smith, and L. Kroc 2006. Multi-scale integrated information and telecommunications system (MIITS): first results from a large-scale end-to-end network simulator, In *Proceedings of the 2006 Winter Simulation Conference*, eds. L. R. Perrone, F. P. Wieland, J. Liu, B.

G. Lawson, D. M. Nicol, and R. M. Fujimoto. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Yan, G., S. Eidenbenz, S. Thulasidasan, P. Datta, and V. Ramaswamy 2010. Criticality analysis of Internet infrastructure. *Computer Networks* 54:1169—1182.

## AUTHOR BIOGRAPHIES

**NANDAKISHORE SANTHI** is a Technical Staff Member at the Los Alamos National Laboratory in the Computer and Computational Sciences Division. Previously he has served as a Professional Consultant for Wireless and Broadband Communication Systems at Synopsys Incorporated and as a Custom Very Large Scale Integrated (VLSI) Circuit Designer at Cypress Semiconductor Corporation. His current research interests include discrete event simulation for infrastructure modeling, algorithms, analysis of problems from communication systems design, algebraic coding theory and combinatorics. He received his M.S. In Electrical Communication Engineering from the Indian Institute of Science Bangalore in 2000 and Ph.D. in Electrical Engineering from University of California San Diego in 2006. He is an active reviewer and has chaired several sessions in major technical conferences. His email address is <nsanthi@lanl.gov>.

**GUANHUA YAN** obtained his Ph.D. degree in Computer Science from Dartmouth College, USA, in 2005. From 2003 to 2005, he was a visiting graduate student at the Coordinated Science Laboratory in the University of Illinois at Urbana-Champaign. He is now working as a Technical Staff Member in the Information Sciences Group (CCS-3) at the Los Alamos National Laboratory. His research interests are cyber-security, networking, and large-scale modeling and simulation techniques. He has contributed about 30 articles in these fields. His email address is <ghyan@lanl.gov>.

**STEPHAN EIDENBENZ** received his Ph.D. In Computer Science from the Swiss Federal Institute of Technology, Zurich in 2000. He obtained a Bachelors degree in business administration from GSBA in Zurich in 1999. He is currently the Deputy Group Leader at the Information Sciences Group (CCS-3) at the Computer and Computational Sciences Division at the Los Alamos National Laboratory, where he leads the socio-technical modeling team. Stephan's current research interests are in wire-line and wireless networking, selfishness in networking, infrastructure modeling, discrete event simulation, and algorithms. Stephan has worked and published in a large number of fields, including approximation algorithms, inapproximability, visibility problems in polygons and terrains, error-modeling in computational biology, and network protocol design, He is also an active reviewer and member of various program committees. His email address is <eidenben@lanl.gov>.