Software Diversity as a Defense Against Viral Propagation: Models and Simulations

Adam J. O'Donnell*, Harish Sethu
ECE Department
Drexel University
3141 Chestnut St.
Philadelphia, PA, USA

E-mail: {adam, sethu}@ece.drexel.edu

Abstract

The use of software diversity has often been discussed in the research literature as an effective means to break up the software monoculture present on the Internet and to thus prevent malcode propagation. However, there have been no quantitative studies that examine the effectiveness of software diversity on viral propagation. In this paper, we study both real (an IPv6 BGP topology) and synthetically generated (an Erdös-Rényi random graph) network topologies and employ a popular metric called the epidemic threshold to measure resistance to viral propagation in the presence of software diversity. We show that one can increase the epidemic threshold of a network even with a naïve, random distribution of diverse software on the nodes of a network. We also show that an algorithm-driven diversity assignment further increases the epidemic threshold. These results confirm the value of strategic topology-sensitive assignment of diversity to improving the tolerance of a network to malcode propagation.

1. Introduction

The field of viral propagation modeling has garnered a great deal of attention in recent years as computer security researchers attempt to find ways of mitigating rapid malcode propagation. A variety of techniques have been suggested which can delay the spread of a worm, including rate-limiting network cards [23], targeted immunization of highly connected nodes [17], and a combination of address blacklisting and content filtering [14]. In complementary work, researchers have been focusing on the software monoculture on the Internet and its relationship to viral epidemics. The value of software diversity to computer security comes from the fact that an attack written for one piece of software rarely works for a different but functionally equivalent software package. By increasing the number of diverse software packages present on the network, the research argues, the chances that an attack will be effective against a randomly selected node will decrease.

The research literature in software diversity suggests that the introduction of different software packages is an effective method of disrupting the activities of an attacker or a worm, particularly one which repeatedly utilizes a pre-written and unchanging attack to compromise machines. However, there have been no quantitative evaluations of the impact of software diversity on malcode propagation in real network topologies. In this paper, we use a popular metric called the *epidemic threshold* [22] to measure a network's resiliency against malcode propagation and study the steady state prevalence of computer viruses in the presence of software diversity. We show, through both modeling and simulation, that even a simple randomized distribution of diverse software packages can increase the epidemic threshold of both real and synthetically generated computer networks. This paper also shows that an algorithm-driven distribution of diverse software packages [15] can further increase the epidemic threshold and serve as an effective method for preventing worm epidemics.

2. Problem Statement

In previous work [15], we showed that the location of diverse software packages on a network is as critical to effectively diversifying a network as the creation of diverse software packages. A visual example of this is provided in Figure 1, where the introduction of an alternative software package is able to minimize the number

* The author's work was supported by the NSF Graduate Research Fellowship and the Koerner Family Fellowship.

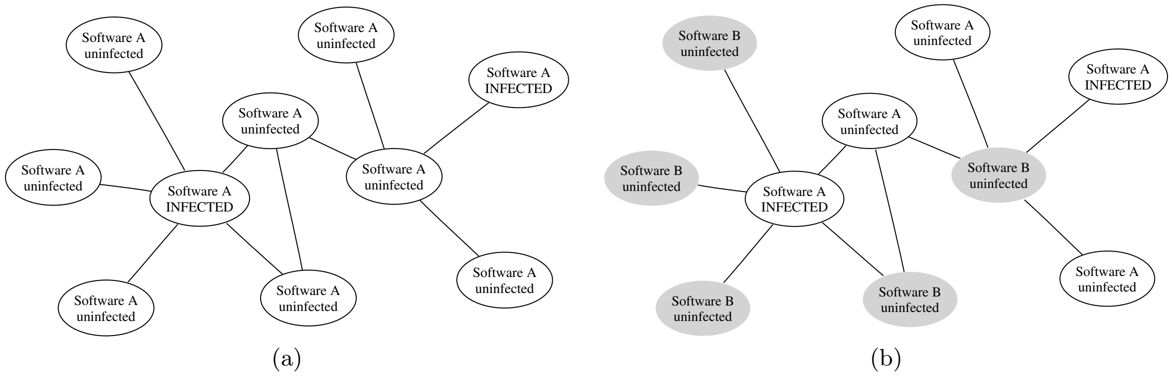


Figure 1. The networks presented above are both infected with a computer virus. The network shown in (a) consists of nodes running the same software package. The nodes in the network shown in (b) are allowed to run two different operating systems. While topologically equivalent, the network shown in (b) provides only a single link for an infectious agent to traverse by reducing the number of vulnerable nodes by half.

of edges across which a virus can traverse. Assuming that different software packages are represented by different colors, this problem translates to that of the distributed coloring of a graph while minimizing the number of monochromatic edges, i.e., the number of homogeneous pairs of neighbors.

While the number of monochromatic edges is an effective metric as an optimization goal, it does not directly express the ability of the diversity assignment algorithm to limit the virulence of a worm. This paper, on the other hand, quantifies the quality of a software diversity assignment by focusing on the effect that network assignments of diverse software has upon the propagation of worms. Given a worm whose rate of propagation from an infected node to each of its vulnerable neighbors is β and the rate at which infected nodes are disinfected is δ, we study the *epidemic threshold*, or the ratio of β/δ below which an infectious agent will burn itself out (i.e., the ratio below which there will be no infected nodes in the network at steady state).

One of the goals of any virus mitigation technique should be to increase the epidemic threshold of the network. In this paper, our goal is to study:

1. The epidemic threshold with a randomized distribution of diverse software packages to nodes in a real network (an IPv6 BGP topology) as well as a synthetically generated (an Erdös-Rényi random graph) network topology.
2. The relationship of the above results to the number of different software packages available to distribute among the nodes.
3. The epidemic threshold on the same networks with a topology-sensitive algorithm-driven distribution of diverse software packages.

Consider a network of computers represented by graph G, a set of diverse software packages C which can be assigned to nodes on the network, and a contagion which can infect only a single software package in C. Assume that the number of software packages available in C is greater than or equal to the chromatic number of the graph $\chi(G)$. If the software packages are randomly distributed to the network, then a portion but not all of the nodes will be rendered immune to the infection. However, if a graph coloring algorithm is used to assign the software in C to the nodes in G, then no edges will be left to spread the infection, and the infection is guaranteed to die out.

This paper is organized as follows. Section 3 describes related work in the fields of software diversity and viral propagation modeling. A generalized analysis of the viral propagation models and the impact of diversity upon them is presented in Section 4. In Sections 4.1 and 4.2, we extend both the statistically derived and graph theoretic viral propagation models to incorporate the impact of random as well as algorithm-driven diversity assignments. We validate these models using simulations of virus propagation on both synthetic and *real* network topologies in Sections 4.1.1 and 4.2.1. The simulations show that the improvement in the epidemic threshold experienced under an algorithm-driven diversity assignment algorithm is significantly higher than that predicted by the bounds generated by our models for real-world graphs. Finally, we present our concluding comments in Section 5.

3. Related Work

The research discussed in this paper is based upon work from two independent but related fields, *software* *diversity* and *viral propagation modeling*. Software diversity research focuses on the creation and distribution diverse software packages to limit the exploitability of a security vulnerability by a worm, known as the *wormability* of a vulnerability [18]. While software diversity focuses on the interaction between a worm and a system at the moment of infection, the field of viral propagation research focuses on the modeling of large scale behavior of worms once they are established in the network.

3.1. Software Diversity

Position papers that assess the inherent value of a heterogeneous population of software packages have been published in both peer-reviewed conferences [24] and in more public forums [7–9, 19]. Literature on the topic contains numerous methods for artificially introducing diversity to software, including the use of source code modification [3, 6, 20], virtual hardware modification [1, 11], and compile time randomization techniques [2, 4, 6]. Just and Cornwell [10] provided a survey of these techniques, and characterized them as a method of breaking the "virtual specification" for attack used by attackers to write worms and viruses. A further classification of diversity techniques is provided by Keromytis and Prevelakis in [13].

In many practical situations, the number of diverse software packages may be insufficient to guarantee that every node is different from every other node on the network. Our previous work [15] showed that careful assignment of a limited set of diverse software packages reduces, and in some cases prevents, an attacker's ability to leverage a single vulnerable system in order to attack additional systems running the same software. In the paper, we equate different software packages to colors on a graph, and then propose a series of distributed graph coloring algorithms for the assignment of diverse software packages, each of which can be used in conjunction with one another to reduce the impact on the algorithm of disinformation from malicious nodes.

3.2. Viral Propagation Modeling

All of the models discussed below are based upon the SIS, or Susceptible-Infected-Susceptible paradigm, where the individual vertices on a graph are either in one of two states: *susceptible* to infection or *infected*. A node moves from the susceptible state to the infected state when an infected neighbor, with the probability β, passes on its contagion. A node moves from the infected state to the susceptible state, independent of its number of neighbors, with the disinfection probability δ. As discussed before, if the ratio β/δ is below the epidemic threshold, the infection will eventually die out.

Kephart and White [12] considered viral propagation on a Erdös-Rényi random graph in an early contribution to the study of computer virus epidemiology. Their assumptions regarding the homogeneity of the nodes in the communication network allowed the authors to model the behavior of an infectious agent using a first order differential equation. A steady state solution for the differential equation is found which provides a bound on the epidemic threshold as a function of the average degree in the graph. They show that the ratio of the infection rate to the disinfection rate must be less than the inverse of the average node degree, $\langle k \rangle$, in order to prevent an epidemic:

$$\frac{\beta}{\delta} < \frac{1}{\langle k \rangle}$$

Pastor-Satorras and Vespignani produced a model which provides insights into the propagation of viruses on graphs with arbitrary degree distributions [16]. Their analysis provides a bound on the epidemic threshold in terms of the node degree's first and second order statistics:

$$\frac{\beta}{\delta} < \frac{\langle k \rangle}{\langle k^2 \rangle}$$

The authors leveraged statistical mechanics to determine closed-form expressions of the second-order statistics of degree distributions for specific classes of graphs. When applied to synthetic graphs which are statistically similar to real world networks, the model predicts that every infection will become an epidemic as the number of nodes tends to infinity. On graphs sampled from real-world data, the number of nodes is finite, and while small, the epidemic threshold is non-zero and can be evaluated numerically.

Y. Wang and others [21] created a discrete time model which is then converted to vector-space notation, which encapsulates the infection state of each node on the network. Using algebraic manipulation, they isolate a system matrix which determines the current infection state based upon the previous system state using a method similar to solving discrete time Markov chains. Spectral decomposition bounds the epidemic threshold of a virus propagating on the network to the inverse of the largest eigenvalue of the graph's adjacency matrix **A**:

$$\frac{\beta}{\delta} < \frac{1}{\max_{\forall i}\{\mu_i(\mathbf{A})\}}$$

Each of the different models examined here have seemingly different methods to determine the upper bound on the epidemic threshold. For the Kephart and White model, the epidemic threshold can be maximized by minimizing the average number of adjacent systems which are vulnerable to a worm. The epidemic threshold will be increased in the Pastor-Satorras and Vespignani model by minimizing the second order statistic while maximizing the first order statistic. The Wang model's epidemic threshold can be maximized by minimizing the largest eigenvalue of the diversified network's adjacency matrix. We show, however, through modeling and simulation, that all of these goals can be met by reducing the number of edges across which a virus can traverse in a diversified network.

4. Viral Propagation and Software Diversity

It is possible to show that, regardless of the underlying viral propagation model, an assignment of software packages to a graph such that the assignment forms a perfect graph coloring will force the epidemic threshold to infinity. Consider a perfect coloring, where there are no edges across which a virus can propagate. The only infected hosts that exist are those which are initially infected by a virus. Because this set cannot increase, the disinfection rate of systems will continually decrease the number of infected systems until all systems are uninfected.

As pointed out in [15], it may not be possible to guarantee that a sufficient number of software systems will be available to perfectly color the network. It would then be more appropriate to assign the limited amount of diversity so as to limit the number of monochromatic edges and thus increase the epidemic threshold. To achieve this goal, we use the COLOR FLIPPING algorithm described in [15]. The distributed algorithm has each node choose an initial software package, or color, and, at random intervals, communicate with their immediate neighborhood of nodes to discover their current color. The node initiating the communication will then switch to the neighborhood's minority software package if it finds a majority of its neighbors are running the same software package.

The introduction of a graph coloring algorithm removes some of the assumptions of randomness that underpin the statistical models discussed, which results in loose bounds on the epidemic threshold on networks colored using the COLOR FLIPPING algorithm. Rather than providing only loose bounds, we examine the effect of algorithm-driven color assignments on the epidemic threshold primarily through the use of simulation.

4.1. Statistical Models

We can consider nodes which run software packages which are different from their neighbor to be relatively immune to attack from their neighbor. Assuming a randomized distribution of diverse software packages, if there are c software packages available for n nodes, it is expected that $n - n/c$ nodes will be relatively immune to the n/c vulnerable nodes.

The effective infection rate, or the rate any given infected node can infect a neighboring homogeneous node, becomes:

$$\beta' = \beta \frac{1}{c}$$

with an epidemic threshold given by:

$$\frac{\beta}{\delta} < \frac{c}{\langle k \rangle}$$

For a given β and δ, the critical number of software packages needed to ensure that a worm infection does not become an epidemic is:

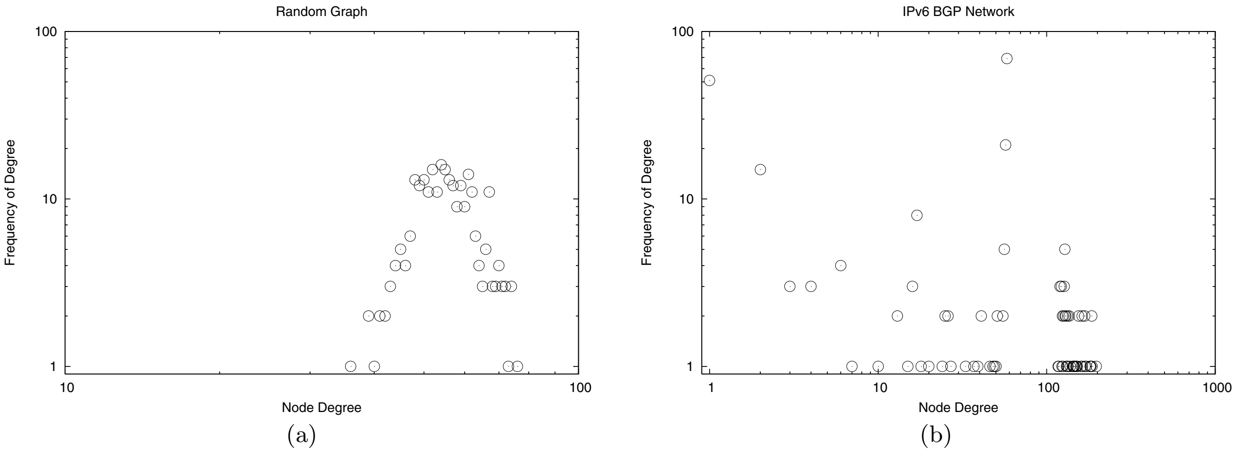
$$c_{crit} = \left\lceil \frac{\beta}{\delta} \langle k \rangle \right\rceil$$

A similar analysis can be done for the Pastor-Satorras and Vespignani model, which shows an increase in the epidemic threshold by a similar factor.

4.1.1. Simulation In order to test the utility of diversity assignments for increasing the epidemic threshold, it is necessary to either generate or measure a network topology for simulation study. Our first network was generated by collecting a list of the BGP peers present in the IPv6 network by accessing the routing table from IPv6 capable Looking Glass routers. A second network was created using an Erdös-Rényi random graph generator. Both graphs contain 266 nodes and approximately 7500 edges. The distribution of the individual node degrees is shown on a log-log scale in Figure 2. While both graphs have similar average degree, the degree distribution for both graphs is dramatically different. The distribution plot of the synthetic graph, shown in Figure 2(a) corresponds to a standard random graph, while the distribution of the sampled graph's topology, shown in Figure 2(b), shows the same self-similar characteristics that have been observed in previous literature [5].

The rest of the simulation studies presented in the paper follow a standard methodology; a single color is tagged as being vulnerable to infection, and the graph is assigned an initial coloring. A high percentage of the nodes assigned the vulnerable color are randomly chosen to be the nodes which initially contain the infection. We experimentally determine the epidemic threshold by progressively changing β relative to a fixed δ until a persistent infection is not seen over numerous simulation runs with both the same initial infection set and with alternate initial infection sets.

The simulation exercises shown in Figures 3(a) and (b) examine the effect that the number of colors has upon theoretically derived and experimentally-



(a) (b)

Figure 2. Plot of the degrees of nodes found in the examined networks versus the frequency of the occurrence of the degree. The graph examined in (a) was constructed from a standard random graph model, and contains 266 nodes and 7448 edges. The graph examined in (b) was sampled from the IPv6 BGP topology, and contains a similar number of nodes and edges.

evaluated epidemic thresholds. For each color, we compute the diversity-aware variants of the Kephart and White (KW) model and the Pastor-Satorras and Vespignani (PV) model presented in Section 4.1 for the random graph and the IPv6 graph, respectively. Additionally, both randomized and algorithm-driven color assignments, based upon the Color Flipping algorithm presented in [15], are performed on the graph for each color count examined.

The data shows that the bound on the epidemic threshold of a randomized coloring provided by the statistical models is below the experimentally determined epidemic threshold. The result allows us to conclude that the epidemic threshold of a diversified network will be higher than the epidemic threshold of a homogeneous network even if diversity is assigned randomly. It is noteworthy that the epidemic threshold is significantly increased by allowing an algorithm to assign diverse software packages to nodes on the network; this leads us to conclude that a planned diversity assignment is a worthwhile undertaking in order to maximize the epidemic threshold of a network.

4.2. Graph Theory Derived Models

In a fashion consistent with Wang's model, we are able to restate the goal of the software assignment in terms of graph partitions and the subsequent eigenvalues of the subgraphs. We denote our software assignment as $f : V(G) \mapsto C, C = \{1, 2, ..., c\}$, where C is the set of available software packages. Let $G_i := G[f^{-1}(i)] : i \in C$, where G_i are the subgraphs induced by color i. Define $\mu_{max}(G_i)$ as the maximum eigenvalue of subgraph G_i's adjacency matrix. Therefore, we wish to find a software assignment f_{opt} where:

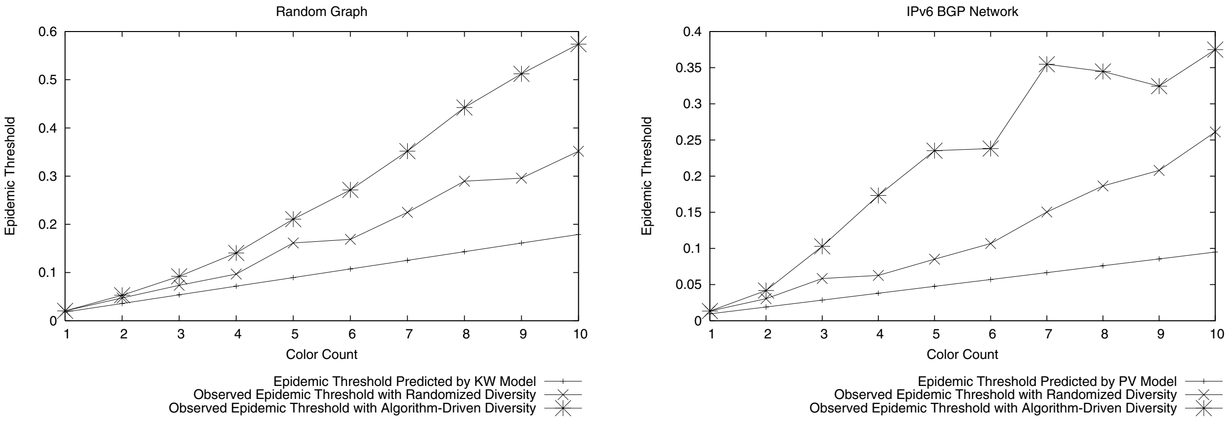
$$f_{opt} = \arg \min_{\forall f} \left\{ \max_{i \in C} \{\mu_{max}(G_i)\} \right\}$$

which minimizes the maximum eigenvalue across all subgraphs induced by each color.

Loose bounds for general graphs and hard bounds on regular graphs can be determined for the largest eigenvalue of the adjacency matrix of a diversified network. Rather than relying upon the loose bounds, we directly measure the eigenvalue of a network which is actively undergoing diversification to predict the epidemic threshold.

4.2.1. Simulation To examine the impact the number of monochromatic edges has upon the epidemic threshold, we simulate a homogeneous network of systems, then allow each system to minimize its number of monochromatic neighbors by executing the Color Flipping algorithm presented in [15]. At each timestep, we compute the epidemic threshold predicted by the Pastor-Satorras and Vespignani model and Wang's eigenvalue model. The Kephart and White model is inappropriate for use with networks using an algorithm-driven diversity assignment as the application of the algorithm to the network removes the homogeneous degree distribution on the network.

Figures 4(a) and (b) show the impact that decreasing the number of monochromatic edges has upon the statistical, eigenvalue-derived, and experimentally



(a) (b)

Figure 3. Comparison of the effect of the number of colors on the experimentally determined epidemic threshold. In both (a) and (b), a graph is assigned either one color for every node, multiple colors via a randomized algorithm, or multiple colors via the described COLOR FLIPPING algorithm. It can be seen in both graphs that the epidemic threshold increases as the diversity-assignment algorithms become progressively more sophisticated.

found epidemic thresholds. It is clear from the simulation studies that reducing the number of monochromatic edges in the network is an extremely effective method of increasing the epidemic threshold. The simulation studies confirm the utility of recomputing the eigenvalue-derived epidemic threshold with each step of the graph coloring operation is an effective method of approximating the epidemic threshold. Furthermore, the experiment shows that decreases in the number of defective edges go hand in hand with increases in the epidemic threshold.

5. Concluding Remarks

While a wide variety of techniques for mitigating rapid malware propagation have been analyzed and simulated using standard virus modeling techniques, the contributions of the software diversity community have not yet been fit into this framework. In this paper, we make the first contributions toward analyzing viral propagation modeling in the presence of software diversity. We use both models and simulations to show that on both simulated and real networks of systems, a naïve, randomized software diversity assignment is able to increase the epidemic threshold. Simulations also show that an algorithm-driven diversity assignment is able to further increase the epidemic threshold beyond that seen with a randomized assignment. These results provide quantitative insight into the impact of software diversity on the tolerance of a network to viral attack.

Acknowledgments: Finally, we would like to thank Dave Richardson of Drexel's CS Department for his comments on the early drafts of the paper, and Madhusudan Hosaagrahana of Drexel's ECE Department for alerting us to this conference.

References

[1] E. G. Barrantes, D. H. Ackley, T. S. Palmer, D. Stefanović, and D. D. Zovi. Randomized instruction set emulation to disrupt binary code injection attacks. In *Proc. of the 10th ACM Conference on Computer and Communication Security*, pages 281–289. ACM Press, 2003.

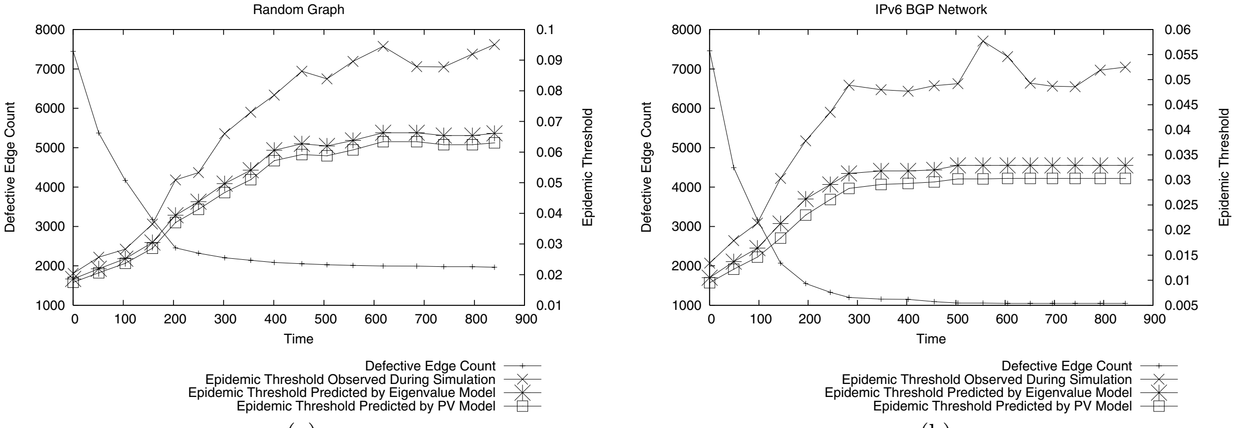
[2] S. Bhatkar, D. C. DuVarney, and R. Sekar. Address obfuscation: An efficient approach to combat a broad range of memory error exploits. In *Proc. of the 12th USENIX Security Symposium*, pages 105–120, Washington D.C., USA, August 2003.

[3] C. Collberg, C. Thomborson, and D. Low. Breaking abstractions and unstructuring data structures. In *Proc. of the IEEE International Conference on Computer Languages*, pages 28–38, Chicago, IL, May 1998.

[4] H. Etoh. GCC extension for protecting applications from stack-smashing attacks, 2004. http://www.trl.ibm.com/projects/security/ssp/.

[5] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *Proc. of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 251–262. ACM Press, 1999.

[6] S. Forrest, A. Somayaji, and D. Ackley. Building diverse computer systems. In *Proc. of the 6th Workshop on Hot*



(a)

(b)

Figure 4. Comparison of the effect of the number of defective edges on the epidemic threshold. In both (a) and (b), the nodes of the graphs all begin at the same color, and the COLOR FLIPPING algorithm from [15] is executed to find a 3-color assignment which reduces the number of monochromatic edges. As the number of monochromatic edges decreases, the experimentally determined epidemic threshold increases beyond what is predicted by statistical models and by the eigenvalue model.

Topics in Operating Systems (HotOS-VI), pages 67–72. IEEE Computer Society, 1997.

[7] D. Geer. Monopoly considered harmful. *IEEE Security & Privacy Magazine*, 1(6):14–16, December 2003.

[8] D. Geer, R. Bace, P. Gutmann, P. Metzger, C. P. Pfleeger, J. S. Quarterman, and B. Schneier. Cyberinsecurity: The cost of monopoly. Tech report, CCIA, 2003. http://www.ccianet.org/papers/cyberinsecurity.pdf.

[9] G. Goth. Addressing the monoculture. *IEEE Security & Privacy Magazine*, 1(6):8–10, December 2003.

[10] J. E. Just and M. Cornwell. Review and analysis of synthetic diversity for breaking monocultures. In *Proc. of the 2nd Workshop on Rapid Malcode*, Washington, D.C., October 2004.

[11] G. S. Kc, A. D. Keromytis, and V. Prevelakis. Countering code-injection attacks with instruction-set randomization. In *Proc. of the 10th ACM Conference on Computer and Communication Security*, pages 272–280. ACM Press, 2003.

[12] J. O. Kephart and S. R. White. Directed-graph epidemiological models of computer viruses. In *Proc. of the IEEE Symposium on Security and Privacy*, Oakland, CA, May 1991.

[13] A. D. Keromytis and V. Prevelakis. Dealing with system monocultures. In *Proc. of the NATO IST Panel Symposium on Adaptive Defense in Unclassified Networks*, Toulouse, France, April 2004.

[14] D. Moore, C. Shannon, G.M. Voelker, and S. Savage. Internet quarantine: Requirements for containing self-propagating code. In *IEEE INFOCOM*, pages 1901–1910, March – April 2003.

[15] A. J. O'Donnell and H. Sethu. On achieving software diversity for improved network security using distributed coloring algorithms. In *Proc. of the 11th ACM Conference on Computer and Communications Security*, pages 121–131, Washington, D.C., October 2004.

[16] R. Pastor-Satorras and A. Vespignani. Epidemic dynamics and endemic states in complex networks. *Phys. Rev. E*, 63(066117), 2001.

[17] R. Pastor-Satorras and A. Vespignani. Immunization of complex networks. *Phys. Rev. E*, 65(036104), 2002.

[18] A. Powell. Internet worms. Tech Report 00727, NISSC, 2003. http://www.niscc.gov.uk/niscc/docs/re-20030805-00727.pdf.

[19] M. Stamp. Risks of monoculture. *Commun. ACM*, 47(3):120, 2004.

[20] C. Wang, J. Davidson, J. Hill, and J. Knight. Protection of software-based survivability mechanisms. In *Proc. of the International Conferece on Dependable Systems and Networks*, pages 193–202, July 2001.

[21] Y. Wang, D. Chakrabarti, C. Wang, and C. Faloutsos. Epidemic spreading in real networks: An eigenvalue viewpoint. In *22nd Symposium on Reliable Distributed Systems*. IEEE Computer Society, October 2003.

[22] Y. Wang and C. Wang. Modeling the effects of timing parameters on virus propagation. In *Proc. of the 2003 ACM Workshop on Rapid Malcode*, pages 61–66. ACM Press, 2003.

[23] M. M. Williamson. Throttling viruses: Restricting propagation to defeat malicious mobile code. In *Proc. of the 18th Annual Computer Security Applications Conference*, page 61. IEEE Computer Society, 2002.

[24] Y. Zhang, H. Vin, L. Alvisi, W. Lee, and S. K. Dao. Heterogeneous networking: a new survivability paradigm. In *Proc. of the 2001 Workshop on New Security Paradigms*, pages 33–39. ACM Press, 2001.