

# Can a Network be Protected from Single-Packet Warhol Worms?

Larry G. Irwin II & Richard J. Enbody  
Department of Computer Science & Engineering  
3115 Engineering  
Michigan State University  
East Lansing, MI 48824-1226  
enbody@cse.msu.edu

## ABSTRACT

Can a network be protected from single-packet Warhol worms [14]? This paper generates and simulates random network environments to answer that question. The research assumes a perfect detection algorithm and varies the time required to perform the identification. Perfect detection alone is not sufficient; it must also be swift in recognizing threats as some cases presented here show that perfect detection offers no noticeable protection. The impact of other network factors on worm propagation and prevention are investigated as well, including: router participation in the prevention scheme, the percentage of routers involved in the traffic passing, and the ability for participating routers to communicate. The results are promising: realistic simulations without communication can protect over 50% of the network. The addition of communication increases that protection to over 80%. The key result is that emerging identification technologies such as LeBrea [31] can be leveraged into viable automated network protection systems against single-packet worms.

**Keywords:** Self-propagating code, worms, routers, Internet.

## 1. INTRODUCTION

Can a network be protected from single-packet Warhol worms [14]? Or do they move so fast that nothing can be done? We began this work expecting that some worms moved too fast for any defense, but found that some characteristics of worms allow one to protect large parts of a network. Since we were interested in the extreme question of whether any defense was possible we use an almost oracle-like detector on a reasonably simulated network for our investigation. Quick detection as worms pass by combined with small worms' use of randomly generated addresses allows one to shut off significant parts of the Internet. We consider a super-efficient worm detector as well as a more reasonable model and then we consider three strategies for fighting worms for each model. The work here is offered as a preliminary set of findings to determine if additional scrutiny is warranted.

Self-propagating code, more commonly referred to as a "worm," is a computer program engineered to place copies of itself on remote machines without the express consent of any of the machines involved. The specifics of how a worm works is outside the scope of this document; the interested reader is directed to [2, 4, 13] for further information. Programs of this nature have existed publicly as far back as 1988 [4, 13]; however, worm epidemics have recently grown in both scope and severity [14].

There have been multiple strategies investigated as possible methods for combating the spread of worms. These methods include: address blacklisting and content filtering [9]; selective shutdown [10]; and traffic throttling [11]. A variety of approaches have been used to study worms which include [26] where Chen, Gao and Kwait extend the simple epidemic model to account for observed real-world behaviors like system death and system patching. Other models, such as the one used in [25] to simulate Code Red, eliminate the issue of topology, yet produce results similar to others. There is a set of simulation work

focused on possible counter-measures to defend against the spread of malicious self-propagating code. Wang, Knight and Elder explore the effect of immunization on various topologies in [29] and Moore, Shannon, Voelkner and Savage evaluate various defenses and parameters in [9]. Moore, et al. utilize mathematical and proprietary simulations to investigate the effectiveness of containment strategies, reaction time and blocking location.

The remainder of this paper is organized as follows. First, section 2 describes how the simulation is structured and operates. Then, Section 3 describes the results for the various content filtering experiments. Section 3 also introduces the difference between the realistic and extreme simulation parameters and examines various types of methods to distribute knowledge among the reacting routers. Section 4 summarizes the results obtained from the experimentation and attempts to explain the observed behavior.

## **2. SIMULATION SOFTWARE**

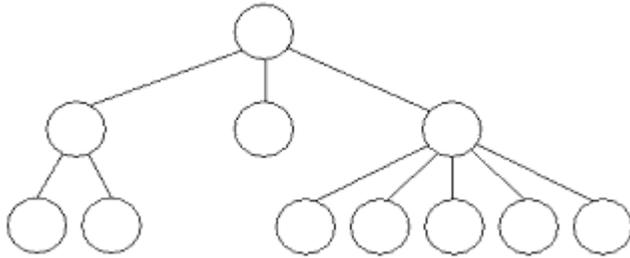
Many methods exist that allow one to simulate networks including NS2 [22], SSFNet [23] and various academic packages [24, 28]. All simulations are close approximations to the true behavior of the actual Internet, but the web's large size and dynamic nature make it an open question as to how to correctly simulate the environment. Worm simulations are debated as well [25,26,27] with models ranging from the biological epidemic model [2] based on how disease spreads within a population to the two-factor worm model [25] where technical aspects are considered including link utilization, system death and patching. However, these simulations are not easily modified to account for routers that have varying behavior; a modification to alter the basic elements of NS2 and SSFNet require intimate knowledge of the framework. The work here is offered as a preliminary set of findings to determine if additional scrutiny is warranted.

### **2.1 Implementation**

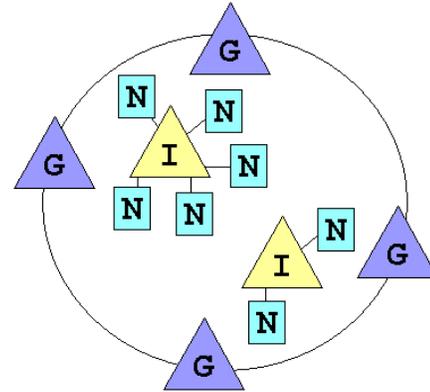
The simulator is written in C++. The framework is a time-step simulation comprised of four main components representing basic elements of a network: autonomous system (AS), router, node and packet. An AS is defined as a collection of IP networks under control of a single entity, typically an Internet service provider [30].

### **2.2 Design**

The software, NetSim, creates random tiered networks (see **Figure 1**), similar in some ways to those networks constructed in [29]. The tiers are primarily based on the ISP structure of the Internet whereby tier-1 Internet service providers (ISPs) lease access to regional ISPs, which in turn offer access to local ISP providers [7]. Instances do arise in the Internet where two ISPs that are not directly next to one another within the hierarchy establish a link between themselves. These peer-connection links are not currently part of the model. A tree of AS objects may be used to simulate the Internet because of the "significant degree of hierarchy" [20]. A sample diagram of how one AS might be composed is shown in **Figure 2**.



**Figure 1: Example AS Tree Structure:** k-ary tree showing how a collection of autonomous systems may be arranged internally to represent a virtual network.



**Figure 2: AS Example** A sample AS object with gateway routers (G), internal routers (I) and nodes (N).

### 2.2.1 Physical Entities

Each **AS object** has a randomly generated set of child AS objects. Also, each AS contains random sets of routers divided into two groups: gateway routers and internal routers.

The two different sets of **routers** are represented by identical software objects. The difference is that the internal routers utilize pointers to child nodes while the gateway routers do not. Both routers have queues that hold packets during simulations and follow identical algorithms pertaining to traffic forwarding. The creation of child nodes for each internal router is handled similarly to the child AS objects discussed previously; each router is initialized with a random number of child nodes.

Perhaps the most important attribute of the routers is the **randomly-assigned reactionary flag** which determines if a router participates in worm detection and prevention. The efficiency of detection is determined by a simulation constant that indicates how many malicious packets must be forwarded before the router recognizes the threat. The learning is assumed to be perfect. Recognizing threats in real-time is an open question [14, 17, 1] but technologies such as LeBrea [31] offer hope that near-perfect detection is around the corner.

**Nodes** are objects within the network that are capable of containing a worm; most likely this equates to a computer. Each node has two main components: a flag and a timer. The flag indicates if the node has been infected and the timer regulates when the node should attempt to spread. There is also a parameter that allows a node to remain pristine for some additional time steps after being initially infected to account for the time required for the worm to completely infest the node. Nodes are randomly infected once the network has been initialized, and before any time steps are started. Initially infected nodes may exist at any depth in the network.

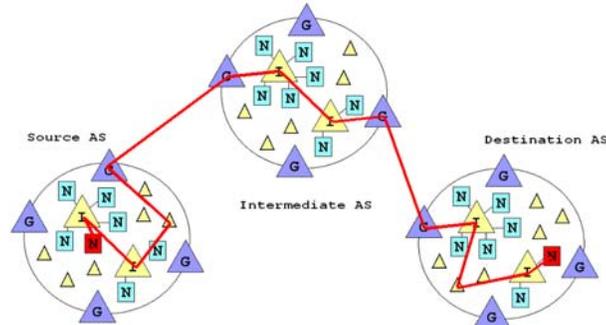
### 2.2.2 Worm Propagation

Once a node succumbed to the worm, the worm attempts to spread further. The strategy of the worm is similar to that of the Slammer worm [18] in that it selects addresses at random. The success ratio of the random selection is controlled by a simulation constant. The rate at which the worm attempts to spread is determined by a timer stored in each node. An infected node sends a packets only if the randomly selected destination is valid.

When a node receives a packet there is another timer that is started that represents the time it takes for the node to be compromised once contact has been made. After this timer expires, the node is marked as infected and its own internal timer is set that determines the rate at which this newly infested node will attempt to spread. At this point the node is infected and will remain so until the simulation is terminated.

### 2.2.3 Packet and Routing Description

Packets are simply small container objects that are passed between routers and AS objects as needed. When packets are created and when packets first enter an AS, the path is determined by the AS object. Placing the routing in the AS allows the simulation to eliminate the issues associated with selecting one routing protocol over another. Furthermore, it makes it easy for the simulation to specify how much of the internal routers within an AS should be involved in transporting packets. **Figure 3** illustrates a sample path taken by a packet assuming that all AS objects contain 100 internal routers and the percentage of participation is 2%.



**Figure 3: Example Route for Packet**

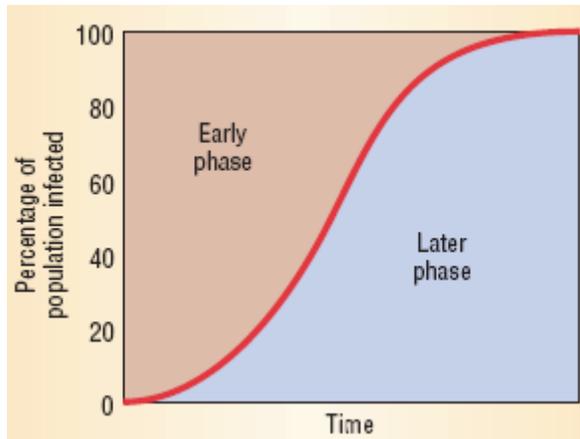
## 3. EXPERIMENTATION

Unless otherwise stated, results presented are the average of 1000 runs of the simulation performed with the identical simulation constants with random seed values from 1-1000. The standard deviation for the extreme simulations regarding network infection percentage averaged roughly 3.5%. Standard deviations for network infection percentages ran under the extreme scenario averaged 3.2%. There is also set of milestones that are recorded for each averaged set of results that provide a set of criteria on which each set of experiments can be measured. Over all simulations, the maximum number of AS and router entities were 256 and 1024 respectively. The actual number of nodes created in the simulation was about 9,000 on average. The k-ary tree of AS objects was limited to a depth of 5 and actual simulations exhibited AS objects reaching maximum depths of 4 or 5. These maximum depths correspond to the various layers discussed in [16]: dense core, transit core, outer core, regional ISPs and customers. The parameters to create the tree directed each AS to randomly generate child AS objects over the range [8, 12]. The number of routers within each AS was inversely proportional to tree depth while the number of nodes increased proportionately with tree depth, per [19]. Routers required two time steps to process packets and worms attempted to spread every five time steps.

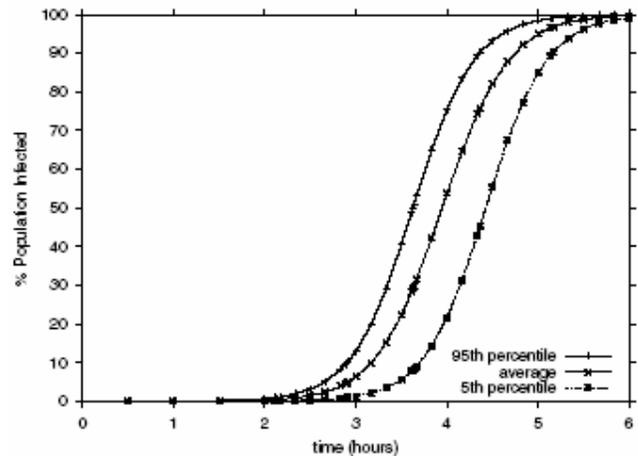
The experiments were conducted under two different scenarios: extreme and realistic. Each scenario shares the simulation settings discussed in the previous paragraph; two settings define the scenarios. The extreme scenario recognizes malicious code in five packets and utilizes 20% of the network to transport packets. The more realistic scenario recognizes the worm in 500 packets and only uses 2% of the internal routers to move packets [7, 31]. Finally, each of these scenarios is conducted under various levels of router participation: 33%, 67% and 100% of the routers throughout the network are deemed reactionary. Simulations ran for a maximum of 1,000,000 time steps, and were shortened when possible to increase the granularity of the recorded statistics. In the interest of space, select results are presented.

### 3.1 Base Case

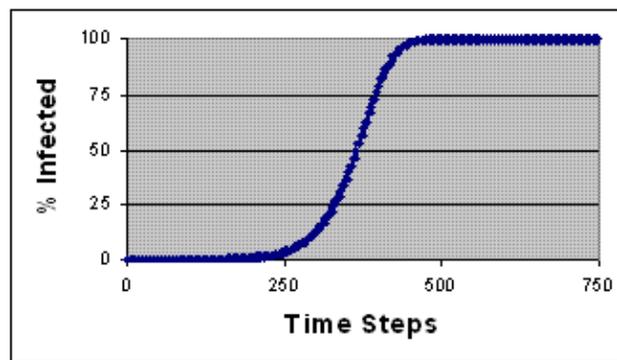
Simulations were conducted without any defense at all to simulate a worm as it would theoretically progress in a modern network. This base case is compared to the results of established models [2, 9] to help assert the accuracy and validity of NetSim. Specifically, **Figure 4** presents the theoretical results using the biological epidemic model discussed in [2] and **Figure 5** shows the some results from [9] concerning the spread of CodeRed. Finally, **Figure 6** is the average result obtained from 1000 simulations using NetSim under the extreme parameter set. The results from the realistic parameter set are nearly identical. The basic curve is similar throughout all the simulations.



**Figure 4: Theoretical Results from a biological epidemic model [2]**



**Figure 5: Results of Code Red-like simulations from [9]**



**Figure 6: Base case results from NetSim using the extreme parameter set.**

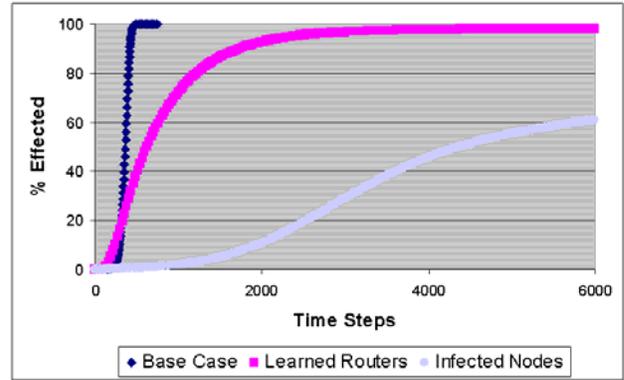
### 3.2 Content Filtering (Packet Dropping)

The first experimentation used a reactionary content filtering scheme. Routers drop malicious packets once the signature was determined (the router became “learned”). The “learned” routers remained “learned.”

**Table 1** and **Figure 7** present values from experiments using the extreme parameter set. **Table 1** lists the final percentage of nodes that became infected while **Figure 7** gives a graphical representation of the progression of the worm as well as knowledge of the worm when one-third of the routers were reacting to the threat. An interesting observation from **Table 1** involves the percentage of routers that had successfully become aware of the worm by the end of the simulation. With less than complete involvement, no more than 40% of the total routers were actively stopping the threat. This is peculiar since the change in overall router awareness increases only 6% while the increase in protection is 46.5%. This behavior implies that strategic locations exist which are more effective at stopping the progression of a worm. The placement of these locations and how they relate to the initial source is not part of this research. Further, the two-thirds and complete participation schemes greatly under-utilize the reactionary router subset. This under-utilization creates unprotected areas that otherwise might have been protected had the routers had a chance to learn during the initial attack.

**Table 1: Max Values for Content Filtering** The table provides the end percentage of infected nodes as well as the routers that had become aware of the worm. The first number in the router column is the percentage with respect to the number of reactionary routers, while the second number is percentage with respect to the number of total routers in the network. These are for the extreme parameter set.

<i>Maximums</i>	Infected Nodes	Learned Routers
<b>drop-33</b>	66.0%	98.6% (32.5%)
<b>drop-67</b>	19.5%	57.6% (38.6%)
<b>drop-100</b>	0.2%	2.4% (2.4%)

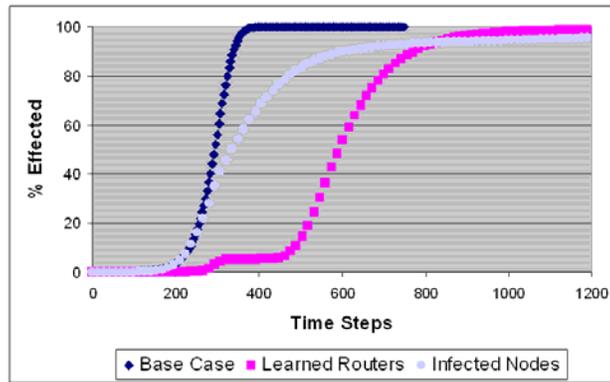


**Figure 7: 33% Reacting Routers Filtering Content.** The graph is obtained under the extreme parameter set and shows that routers become aware of the invasion very quickly. However, even after nearly all possible routers are reacting, the worm continues to spread throughout the network.

**Figure 7** illustrates the case where 33% of the routers were allowed to learn and react. The infection curve is flattened dramatically in comparison to the base case simulation. One factor contributing to the vast difference is the efficiency in the detection. The routers learn of the infection much sooner than the worm progresses. In fact, the knowledge about the worm is faster in the initial stages than even the base case! Despite the nearly immediate recognition, the worm continues to spread further throughout the network after almost all possible reactionary routers are actively dropping packets. This phenomenon implies that there is a maximum protection possible with one-third participation and that maximum is roughly 34% protection (67% infection). The increase in infestation in the face of the reactionary subset completely aware can be explained by the set of routers that are not filtering content. The worm is exploiting these holes, however minute they may be, and is continuing to propagate.

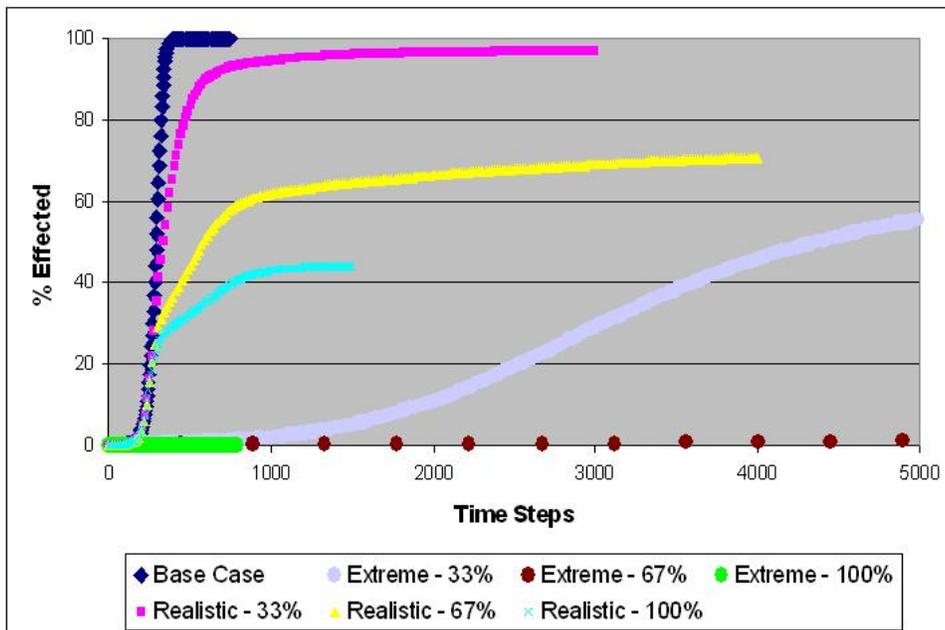
The results from the experiment seem exceptional; that such a remarkably high percent of the network was effectively quarantined as a result of purely reactionary defense. The performance is partially a result of the assumptions made regarding the hypothetical detection algorithm. Perfect recognition in five packets (five packets that had valid destinations in this framework) is highly unlikely given current technology. Therefore, the more realistic parameter set, with its detection requiring 500 packets, is considered next.

**Figure 8** shows the results of simulations under the realistic parameter set with one-third participation. The worm manages to invade nearly the entire network; achieving a final infestation of 97%. The worm exhibits a rate of infestation similar to the base case with slight abnormalities corresponding to the increase in knowledge about the worm among the routers. The active routers inhibit the worm’s propagation and the curve is altered. Despite the effect, the worm still manages to basically infect all nodes. An interesting characteristic of **Figure 8** involves the rate at which the routers learn of the threat. There is an initial increase followed by a plateau of zero advancement. Then, the routers begin to learn of the threat more quickly. These actions arise from the routers that are near the worm and those that are further away. The local routers are more likely to forward a packet and thus more likely to obtain enough information to acquire the signature. Once that initial source alerts its immediate neighbors, those routers close to the source cut-off much of the propagation from that point. However, the worm had likely already advanced to some far reaches of the network and those routers close to the secondary infestations must be given time to learn about the worm themselves.



**Figure 8: 33% Reacting Routers under Realistic Parameters Performing Content Filtering.** The network is not sufficiently defended as the worm infection swells to nearly 100%. The reactors first respond to the initial source, but it is too late to avoid a massive outbreak.

**Figure 9** shows all six infection curves (both simulation parameter sets and three reactionary participation levels) in addition to the realistic base case. The realistic simulations all conclude within the time frame but 33% and 67% extreme simulation results extend beyond the limits of the graph. The graph clearly shows how all three realistic simulations diverge from the base case curve at the same point (divergence point) and have varying degrees of growth from that point forward. The post-divergence rate flattens proportional to the participation of routers.

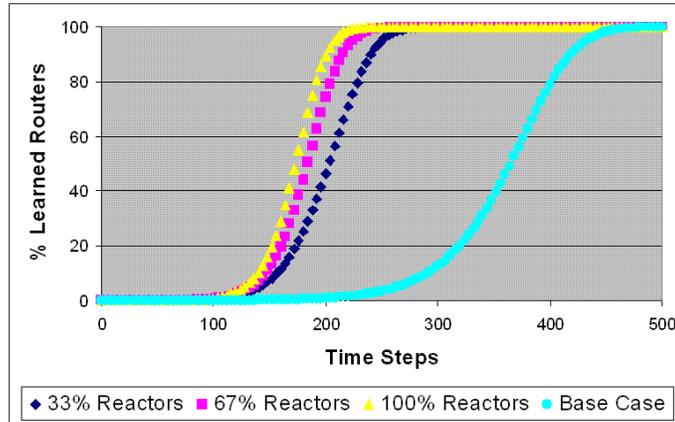


**Figure 9: Content Filtering Infection Curves** All 6 curves associated with basic content filtering with the realistic base case.

### 3.2.1 Fighting Fire with Fire

Based on the characteristics of **Figure 8**, routers were altered to include intra-router communication. This communication is henceforth referred to as “signaling”. Signals cause reactionary routers that have not

learned the signature of the worm to immediately become learned. The initial version of signaling has the routers mimicking the behavior of an infected node—the router will randomly signal other routers after learning the signature of the worm. Randomly selected routers may or may not be reactionary routers. A simulation constant determines the rate that the router sends signals. The rate is set equal to that of the worm.



**Figure 10: Reactor Knowledge Propagation:** knowledge spread faster than the worm.  
(extreme parameter set)

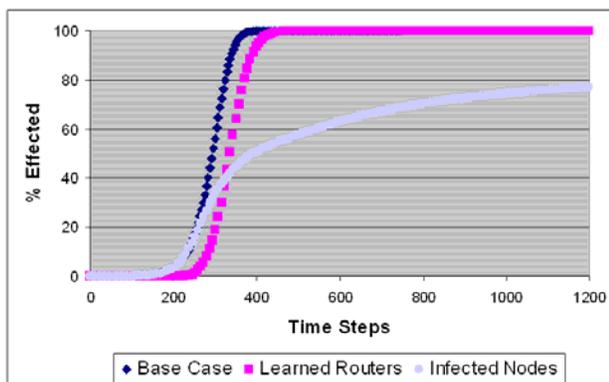
**Figure 10** illustrates how knowledge of the infection can spread faster than the worm itself. In the figure, the base case curve is presented in conjunction with the three participation levels conducted under the extreme parameter set. The knowledge spreads faster than the worm. In fact, the knowledge spread in roughly half the time required for the base case. Interestingly, the more reactionary routers there are, the quicker all reactionary routers learn of the signature. True, there are more routers that must acquire the knowledge, but there are that many more signaling as well which results in more efficient dissemination.

The increase in knowledge dissemination directly impacts the level of protection. **Table 2** presents the final infestation levels achieved by the simulation for the various experiments discussed thus far. The table shows signaling provides a clear improvement under the realistic scenarios. The extreme scenarios exhibit some gains from signaling but are not as notable.

**Table 2: Maximum Infestation Values** Maximum infestation percentages for the experiments discussed thus far: content filtering (Drop) and Fire With Fire.

<i>Max Infested</i>	Realistic		Extreme	
	Drop	Fire With Fire	Drop	Fire With Fire
<b>33% Reactors</b>	97.0%	79.3%	66.0%	65.0%
<b>67% Reactors</b>	70.7%	46.8%	19.5%	13.0%
<b>100% Reactors</b>	43.9%	19.5%	0.2%	0.1%

As seen above in **Table 2**, the experiments were conducted under the realistic parameters as well. These experiments investigate the plateau behavior exhibited in **Figure 8** and intend to show that communication increases knowledge propagation. The corresponding Fire With Fire experiment is shown below in **Figure 11**. Eventually, the worm still infects most of the network (79% infected), but it is a considerable improvement over the earlier results. Also, the plateau that was once present in the knowledge curve has been eliminated and knowledge flows throughout the subnet much faster.



**Figure 11: 33% Reacting Routers under Realistic Parameters Performing Fire With Fire**  
 Reacting routers quickly respond to the worm and significantly alter the worm's progress.

The more realistic model exhibits the benefits of sharing information between reactionary routers. In the extreme case examined earlier, information among the routers spread much more quickly than the worm, and as a result, the routers had learned nearly all that they could before the worm started its aggressive progression into the network. The more realistic models, on the other hand (**Figure 8** and **Figure 11**), show how communication between routers can alleviate the delay in the reactionary curve (**Figure 8**; time 300 through 450). While this approach does successfully increase the preventative abilities of the reacting routers, it does so at the possible expense of network resources and has no means with which to stop signaling their peers: following this model, routers will continue to signal *ad infinitum*. A model that allows reactionary routers to successfully signal its peers and has an automatic mechanism to control the number of signals being generated is essential to a functional defense system.

### 3.2.2 Intelligent Signaling

In response to the two main concerns with the Fighting Fire With Fire signaling (network strain and infinite signaling), we created intelligent signaling. To address network strain, routers will only signal when they drop a malicious packet. Since this is a one-to-one exchange, the strain on the network is no worse than had the worm spread unbounded. Similarly, the infinite signaling is addressed by that design decision as well. Since routers only signal in response to malicious packets, the rate will vary depending on the worm. When the worm is very active, there will be many signals generated; accordingly, when the worm is basically eliminated, the routers will automatically cease from signaling about the threat. Finally, intelligent signaling assumes that reactionary routers are aware of other reactionary routers and thus randomly selects a peer capable of actively defending the network. Similar overlay networks of volatile nodes exist currently as peer-to-peer networks [6, 15].

The impact on knowledge dissemination is examined first. **Figure 12** presents the curves of knowledge propagation among routers as well as percentage of nodes infected for realistic simulations with one-third of the routers participating. The graph clearly shows that the knowledge of the worm has spread even faster than with generic signaling. The increase in knowledge dissemination can be explained by the focused random selection. The Fight Fire With Fire signaling protocol allowed routers to signal devices that could not properly receive the data, thus wasting an attempt to signal a peer. The change in knowledge progression also results in a slight improvement in maximum infestation (see **Table 3**) as well as an infestation curve that has flattened in the later stages (see **Figure 8** and **Figure 11**). These improvements come along with the positive change that network resources are not as strained as they would have been under the Fire With Fire system.



**Figure 12: 33% Reacting Routers under Realistic Parameters Performing Intelligent Signaling**  
 The dissemination of information between reacting routers is focused and thus spreads more quickly and results in a greater percentage of network quarantined.

The final infestation percentages for all the experiments presented in this paper are summarized in **Table 3** below. The benefits of communication are most apparent under the realistic scenarios. The extreme environments simply recognize the infection so quickly on their own that communication offers only a slight improvement. It is noteworthy that the detection is so quick in the extreme case that it appears to limit its own ability to notify other routers of the threat. Since signals are generated only when a malicious packet is dropped, too few infections results in minimal signals. The extreme case quarantines sections of the network so quickly that there are not enough signals as a result of worm packets to alert appropriate routers of the worm. As a result, intelligent signalling actually performs worse than Fire With Fire under the extreme conditions.

**Table 3: Maximum Infestations for extreme scenario.** The values represent the maximum infestation percentages obtained under realistic and extreme scenarios.

<i>Max Infested</i>	Realistic			Extreme		
	Drop	Fire With Fire	Int. Signaling	Drop	Fire With Fire	Int. Signaling
<b>33% Reactors</b>	97.0%	79.3%	72.5%	66.0%	65.0%	65.3%
<b>67% Reactors</b>	70.7%	46.8%	43.4%	19.5%	13.0%	13.8%
<b>100% Reactors</b>	43.9%	19.5%	17.3%	0.2%	0.1%	0.1%

Signaling has shown to be one possible solution to actively alert peer routers of the presence of malicious code. Quite surprisingly, the speed at which signaling is sent throughout the network is not as important as one might suspect when the speed of the routers recognition is sufficiently fast. On the other hand, if the recognition is not that quick, then signaling allows for the set of first responding routers to alert the others. However, even after all routers have been alerted, there may be unprotected paths as long as there are non-reacting routers present in the network. Lastly, the intelligent signaling has shown promise by focusing the signals to known reactors.

#### 4. SUMMARY & CONCLUSIONS

The paper has developed a simulation framework known as NetSim to perform a series of experiments on networks to evaluate possible measures to protect networks from single-packet Warhol worms. The framework was run without any special enhancements to simulate a modern network without any real protection. The results were reasonably similar to that of other work published and considered suitable for this study.

The first approach utilized a purely reactionary system whereby routers would learn a worm signature and then drop packets locally. Routers did not concern themselves about the welfare of other routers within the network. The more realistic experiments demonstrated that perfect detection is not enough since the network was shown to totally succumb to the worm.

The second set of experiments included communication between routers and modeled this additional communication after the propagation of worms. This scheme resulted in an increase of quarantine from the defense system; however, it came at a price. The network would be burdened with the excessive traffic from nodes as well as routers. Further, the routers have no automatic stopping mechanism, so routers would theoretically send signals forever.

The final set of experiments introduced “intelligent signaling” where a reactionary router is aware of its peers and selects one from that list to send a signal. Under the intelligent signaling strategy, signals are only generated when a malicious packet is dropped. For the two parameter settings, realistic and extreme, the improvement over Fire With Fire was noticeable and functionally equivalent, respectively.

The research here emphasizes the relentless nature of worms. The phenomenon is evident through the work. The worm continues to spread regardless that the complete reactionary subnet of routers is aware of the problem. The worm will find a path through non-reactionary routers and accomplish its tasks.

This research concludes that there is a theoretical point at which recognition and information sharing can successfully stop a worm from completely occupying a network—yes a network can be protected. The level of protection is obviously directly related to the percentage of routers that are capable of providing protection. If there are holes in the network, the worm will eventually find them. The goal is to provide information as quickly and efficiently as possible so that those areas with the option of quarantine can take appropriate action and set up defenses. A focused distribution of knowledge does provide some additional level of protection and results can be obtained with as little as one-third participation.

## 5. REFERENCES

- [1] P. Akritidis, K. Anagnostakis, and E.P. Markatos: “Efficient Content-Based Detection of Zero-Day Worms,” Proceedings of the International Conference on Communications (ICC 2005), Seoul, Korea. May 2005.
- [2] T. Chen, J-M. Robert. “Worm Epidemics in High-Speed Networks,” IEEE Computer. June 2004
- [3] Cygwin Website. <http://cygwin.com>
- [4] M. Eichin and J. Rochlis. With Microscope and Tweezers: An Analysis of the Internet Virus of November 1988. Massachusetts Institute of Technology, Boston. 1988.
- [5] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-level traffic measurements from the Sprint IP backbone," IEEE Network Magazine. Nov 2003.
- [6] Gnutella Website. <http://www.gnutella.com>
- [7] J. Kurose and A. Ross, Computer Networking: A Top-Down Approach Featuring the Internet, 3rd Edition. Boston, Massachusetts, USA: Addison Wesley. 2005.
- [8] P. Owezarski, "Does IPv6 improve the scalability of the Internet ?", Joint Internal Workshops on Interactive Distributed Multimedia Systems and Protocols for Multimedia Systems (IDMS/PROMS'2002), Coimbra, Portugal. November 2002.
- [9] D. Moore, C. Shannon, G. Voelker and S. Savage. “Internet Quarantine: Requirements for Containing Self-Propagating Code,” Proceedings of the 2003 IEEE Infocom Conference, San Francisco, CA. April 2003
- [10] K. Patch. “Selective Shutdown Protects Nets,” [http://www.trnmag.com/Stories/2004/082504/Selective\\_shutdown\\_protects\\_nets\\_082504.html](http://www.trnmag.com/Stories/2004/082504/Selective_shutdown_protects_nets_082504.html). August 2004.
- [11] P. Roberts. “HP Shelves Virus Throttler”. <http://www.pcworld.com/news/article/0,aid,117531,00.asp>. August 2004
- [12] S. Savage. Personal communications. October 2004.
- [13] E. Spafford, “The Internet Worm Program: An Analysis”, Purdue Technical Report CSD-TR-823. Department of Computer Sciences Purdue University, West Lafayette, IN. November 1988.

- [14] S. Staniford, V. Paxson and N. Weaver, "How to Own the Internet in Your Spare Time", Proceedings of the 11th USENIX Security Symposium, San Francisco, CA. August 2002.
- [15] I. Stoica, R. Morris, D. Karger, M. Frans Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications," ACM SIGCOMM 2001, San Deigo, CA. August 2001. pp 149-160.
- [16] L. Subramanian, S. Agarwal, J. Rexford and R. H. Katz, "Characterizing the Internet Hierarchy from Multiple Vantage Points," Proceedings IEEE INFOCOM 2002, June 2002. pp. 618-627.
- [17] Worm Blog. <http://www.wormblog.com>
- [18] D. Moore, V. Paxson, S. Savage, C. Shannon, N. Weaver. "Inside the Slammer Worm," IEEE Security and Privacy, July 2003.
- [19] K. Yokum, E. Eagle, J. Degeys, D. Becker, J. Chase, and A. Vahdat. "Toward Scaling Network Emulation using Topology Partitioning," in Proceedings of MASCOTS 2003. October 2003.
- [20] H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. "Network topology generators: Degree-based vs structural," In ACM SIGCOMM. August 2002
- [21] S. Floyd, V. Paxson. "Difficulties in Simulating the Internet," IEEE/ACM Trans. on Networking Vol 9. Aug 2001.
- [22] NS2 Website. <http://www.isi.edu/nsnam/ns/>
- [23] SSFNet Website. <http://www.ssfnet.org>
- [24] High-performance Computing & Simulation Research Lab Website, University of Florida. <http://www.hcs.ufl.edu/>
- [25] C. Zou, W. Gong and D. Towsley. "Code Red Worm Propagation Modeling and Analysis," in 9th ACM Conference on Computer and Communication Security. November 2002.
- [26] Z. Chen, L Gao, and K. Kwiat. "Modeling the Spread of Active Worms," in IEEE INFOCOM. 2003
- [27] D. Moore, Paxson, Savage, et al.. "Inside the Slammer Worm," IEEE Security and Privacy, 1(4):33-39. July 2003.
- [28] INDEX Group Website. <http://lion.cs.uiuc.edu/overview.html>
- [29] C. Wang, J. C. Knight and M. C. Elder. On Computer Viral Propagation and the Effect of Immunization. Proceedings of 16th ACM Annual Computer Applications Conference. New Orleans, LA. 2000.
- [30] Wikipedia "Autonomous System (Internet)" entry.  
[http://en.wikipedia.org/wiki/Autonomous\\_system\\_%28Internet%29](http://en.wikipedia.org/wiki/Autonomous_system_%28Internet%29).
- [31] T. Liston. "La Brea," <http://hackbusters.net/LaBrea/>.
- [32] N. Weaver, Paxson. "A Worst-Case Worm," Proc. 3rd An. Wksp on Economics & Info. Security(WEIS04) 5/2004.