Detecting Smart, Self-Propagating Internet Worms

Jun Li and Shad Stafford
Network & Security Research Laboratory
University of Oregon
{lijun, shad}@cs.uoregon.edu

***Abstract*—Self-propagating worms can infect millions of computers on the Internet in just several minutes. Although there are already many existing worm detectors, none of them systematically consider the countermeasures from worm authors, leaving them potentially ineffective against smart, evasive worms. We therefore revisit worm detection in this paper. We treat worm detection as an arms race, and study how to most effectively detect not only classic worms (i.e. worms that do not have the knowledge of worm detectors), but also evasive worms that know the worm detector in place, know its configurations, and can even adjust their scanning rate by observing legitimate traffic. We describe our design of a new worm detector called SWORD, conduct extensive experiments using realistic trace with different parameters of worms, and demonstrate that SWORD is superior to existing detectors for detecting both classic and evasive worms.**

Keywords*—*Internet worm; smart worm; worm detection; behavior-based worm detection

I. Introduction

Worms can spread themselves and infect millions of hosts on the Internet in just several minutes. Even though there have been fewer outbreaks of computer worms in recent years, they continue to pose a severe threat to the security of the Internet. In fact, the ground for worms to spread in is more fertile than ever. The number of Internet-capable devices continues to rise at a stunning rate, and each of these devices is capable of running a diverse range of user-installable software that can be vulnerable to malicious attacks. Very often these attacks can even penetrate firewalls or NAT boxes if they exist. History also shows that prior to the sudden occurrence of large-scale worms, there can be a relatively long lull without much worm activities. It was more than a decade between the Morris worm in 1988 and a big wave of many devastating worms in early 2000's.

There are many existing worm detectors. These detectors usually assume that worms have no knowledge of a worm detector in place, much less their configurations, and do not observe legitimate traffic in place and adjust their scanning rate accordingly to stay undetected. Such an assumption would be fine if the objective is only to detect *classic* worms that agree with the assumption. Unfortunately, it is often dangerous to make such assumptions about malware. Ideally, a worm detector should continue to function effectively even if a worm is smart and evasive. In the same way that we cannot assume worms are forever gone, we cannot assume worms will just wait to be caught by one of the existing detectors without taking any evasive actions.

We revisit worm detection in this paper. We focus on worms that propagate by themselves, treat worm detection as an arms race, and propose a new worm detector called **SWORD** (Self-propagating Worm Observation and Detection). Unlike most existing detectors that relied on all sorts of behavior to detect worms, SWORD is focused on the *fundamental* behavior of worms and is hard to avoid. As a self-propagating worm is defined as code that scans the network to find and infect new hosts, we believe the only truly fundamental behavior of worms is that of connecting to new destinations. Behavior-based detection systems that do not focus on this one fundamental behavior can be evaded successfully by sufficiently smart worms.

SWORD's working mechanisms are simple and novel. It includes two main modules in detecting violations that a worm will cause in connecting to new destinations, and these two modules complement each other: If a worm does not wish to violate one module when connecting to somewhere, it will inevitably violate the other, leaving little space for a worm to breathe and forcing it to slow down or freeze.

We have designed an experimental framework for evaluating various behavior-based worm detectors. We measured the performance of SWORD and compared it with other detectors. We found that SWORD can not only effectively detect classic worms, *i.e.*, worms that do not have the knowledge of worm detectors, but also evasive worms, *i.e.*, worms that actually do. It significantly outperforms all other detectors. For example, while it is as competitive as the TRW detector in most effectively detecting classic worms, it outperforms TRW by a factor of 60 in most environments in detecting evasive worms (*i.e.*, an evasive worm against TRW can scan 60 times faster than an evasive worm against SWORD).

II. Background and related work

A self-propagating worm running on a host will actively scan the network (or the entire Internet) that the host is connected to and look for new victims to infect. A worm can employ a variety of scanning mechanisms, including random, local preference, sequential, permutation, topological, and hit-list scanning [1]. It infects a remote host by gaining sufficient privileges (typically by exploiting a flaw, such as buffer overflow, in software running on the remote host) to copy itself to, and then execute itself on, the remote host.

There have been a variety of worm detection systems proposed, either host-based or network-based. Host-based detection uses information available at the end-host, and example techniques include buffer overflow detection, correlating network data to memory errors, and looking for patterns in system calls (*e.g.*, [2], [3], [4], [5]). But since host-based detection requires deployment on every host to detect if a host is infected, network-based detection became more desirable with less overhead to install and maintain. Network-based systems

usually only needs a single deployment location—such as a network gateway—to protect an entire network. Network-based detection mainly includes (1) content-based detection which observes the content of network traffic to look for byte patterns that match the signature of a worm (*e.g.*, [6], [7], [8], [9], [10]), and (2) behavior-based detection which observes the network behavior of end hosts and identify patterns that are indicative of the presence of a worm. Because content-based detection is less capable against zero-day or polymorphic worms and can incur a high overhead to inspect traffic payload, we focus on behavior-based detection in this paper.

Existing behavior-based worm detection has focused on various types of traffic behaviors, including: how the outgoing connections from a host correlate to the incoming connections to that host, how the connection failure patterns of a host deviate from normal, and what a host's pattern of visiting destinations looks like. As we will need to compare our detector SWORD against state-of-the-art behavior-based worm detectors, we now summarize these detectors below. We chose the same set of detectors used in comparing the performance of behavior-based worm detectors against classic worms [11].

DSC [12] detects a worm by correlating an incoming connection on a given port with subsequent outgoing connections on that port. If the outgoing connection rate exceeds a threshold established during training, the alarm is raised.

TRW [13] identifies a host as worm infected if its attempts to connect to new destinations result in many connection failures. The basic idea is that a worm-infected host that is scanning the network randomly will have a higher connection failure rate than a host engaged in legitimate operations. Note TRW is based on the *rate* of connection failures rather than the total number of connection failures, and it only counts connections to new destinations (not counting repeated connections to a single destination). Even with the IPv4 address space getting closer to complete allocation, the majority of addresses will not respond to a connection attempt on any given port. Randomly targeted connections (as in worm scanning) will likely fail.

The multi-resolution approach [14], which we refer to as MRW, assumes that when there is no worm the growth curve of the number of distinct destinations over time is concave, but not so when a worm is present since worm scanning will lead to many destinations. This can be leveraged by monitoring over multiple time windows with different thresholds for each window. If the number of new destinations for a host within a given window exceeds the threshold, the alarm is raised.

The Protocol Graph detector [15], which we refer to as PGD, is targeted at detecting slowly propagating hit-list or topologically aware worms. It works by building protocol-specific graphs where each node in the graph is a host, and each edge represents a connection between two hosts over a specific protocol. It assumes that during legitimate operation over short time periods, the number of hosts in the graphs is normally distributed and the number of nodes in the largest connected component of each graph is also normally distributed. During a worm infection, however, both numbers will become abnormal, thus indicating the presence of a worm.

RBS [16] measures the rate of connections to new destinations, similar to MRW. It assumes that a worm-infected host contacts new destinations at a higher rate than a legitimate host does. It measures this rate by fitting the inter-arrival time of new destinations to a exponential distribution.

TRWRBS [16] combines the TRW and RBS detectors into a unified scheme, and observes both the connection failure rate and the first contact rate. It performs sequential hypothesis testing on the combined likelihood ratio to detect worms.

All aforementioned detectors are focused primarily on classic worms, without considering the countermeasures that a smart, evasive worm may employ. Stafford and Li have also evaluated and compared their performance [11], but only against classic worms. Assuming that worms will behave in this fashion may lead to some detectors appearing to perform well, despite the fact that they could be easily evaded by a worm with a more sophisticated behavior pattern.

III. The SWORD Detector

A. Overview

The definition of a self-propagating network worm is code that scans the network to find and infect new hosts. With this definition, the only truly fundamental behavior of worms is that of *connecting to new destinations*. Behavior-based detection systems that do not focus on this one fundamental behavior can be evaded successfully by sufficiently smart worms (which will be validated with the results in Section V). Therefore, the one behavior that we must include to improve worm detection is that of visiting destinations. Are there techniques for detecting anomalous destination visiting patterns that have not been explored yet and that could help to reduce the effective scanning rate of worms?

We depart from existing approaches by focusing on what is essential for a worm to propagate, and devise a new worm detector, SWORD, with superior performance characteristics to the detectors described in Section II. It is positioned at the gateway of a protected internal network as a monitor, as shown in Figure 1. It includes two main modules: a Burst Duration Detector (BDD) that encompasses a burst detection algorithm to prevent fast worm scanning, and a Quiescent Period Detector (QPD) that ensures the quiescent periods in network activity do *not* disappear because of constant worm scannings. A key innovation here is that these two modules complement each other: If a worm does not wish to violate one module when connecting to somewhere, it will inevitably violate the other, leaving little space for a worm to breathe and forcing it to slow down or freeze. Finally, rather than applying a single threshold to an entire network of hosts, SWORD employs a clustering method to group hosts based on their recent activity profile and establish different thresholds for different groups of hosts.

Below we describe BDD, QPD, clustering, and SWORD itself in detail.

B. BDD: Preventing Fast Scanning via the Burst Duration Detector

The most fundamental behavior of self-propagating network scanning worms is the behavior of contacting new destinations seeking new victims to infect. This behavior simply cannot be avoided by a worm that is looking to propagate. So, to detect a worm one should look for anomalies in the rate at which new destinations are contacted, *i.e.*, the rate of *first-contact connections*. The key is then to determine whether or

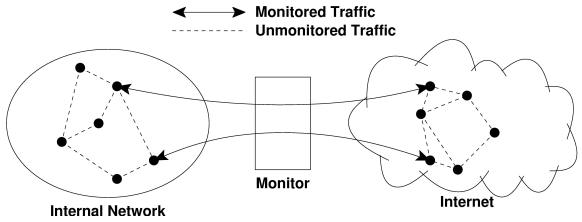


Fig. 1: **SWORD detector running as a traffic monitor at the gateway of a protected network.**

not a host is making first-contact connections at a rate faster than usual.

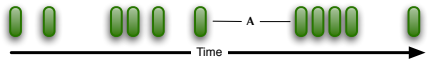
Two previous detectors relied on heuristic of this flavor: the MRW detector and the RBS detector. However, they both have their deficiencies. The MRW detector counts the number of first-contact connections in a series of time windows of different length (hence the "multi-resolution" in the method's title), but it only uses a relatively small set of windows, typically fewer than 10. An intermediate window size might produce a detection window that would detect a worm more quickly than the bigger or smaller sizes in use, but due to the limited number of windows, MRW cannot take advantage of this. RBS, on the other hand, computes a threshold for every different window size, and it uses the number of connections instead of time to describe the window. However, it suffers from sub-optimal thresholds, and thus a poor performance even against classic worms (Section V-A). RBS attempts to fit a single curve to the distribution of inter-connection intervals and uses this curve to generate the thresholds, but in practice the distribution does not map well to a single curve.

BDD avoids the drawbacks in the MRW and RBS approaches. Rather than using fixed time-limited window sizes like MRW, we use RBS's method of creating a window for every different size of connection burst. There is a threshold for a 2-connection burst, a 3-connection burst, a 4-connection burst, and so on, up to a maximum burst size. We also introduce a training phase, during which we measure multiple different durations observed for each burst size and base our threshold on the minimum duration observed for a given burst size. This has the advantages of giving us a threshold for every different size burst (a large number of window sizes), while allowing for a more complex distribution of inter-connection interval times (more accurate thresholds).

The potential drawback to this new method is greater overhead for storing different thresholds and greater computational requirements for examining a recent connection history to determine if it violates any of the thresholds. However, a truism is that computational power and storage space are constantly increasing, and we feel that this additional load is relatively inconsequential.

C. QPD: Ensuring Quiescent Periods via the Quiescent Period Detector

A normal host will exhibit regular *quiescent* periods where it does not make any first-contact connections. In other words, legitimate traffic is typically bursty, with first-contact connections occurring in groups and quiet periods between them. Figure 2(a) shows an example pattern of legitimate connections. Point **A** in the figure shows a quiescent period with no worm traffic, followed by a burst of connections.



(a) Legitimate Connections

(b) Legitimate Connections Plus Classic Worm Connections

(c) Legitimate Connections Plus Rate-Adaptive Worm Connections

Fig. 2: **Examples of observed connections over time**

Below we first see how the presence of a worm may affect a host's quiescent periods, especially if it is adaptive and may evade BDD, and then describe our QPD module.

After a worm infects a host and tries to spread itself, if it scans at a fixed rate, it will make connections during the middle of a legitimate burst, which will raise the overall observed connection rate from the host. Figure 2(b) shows the legitimate traffic with the addition of classic worm traffic. Point **B** indicates a spot of increased connection rate due to the worm connections adding to the burst of legitimate traffic. If BDD is in place, it then can detect the worm.

The worm, however, could be adaptive and avoid this additive effect. Specifically, the worm can dynamically adjust its first-contact rate so that it is always lower than the detection threshold. If the host makes bursts of legitimate first-contact connections, the worm can simply slow down to keep from adding too many its connections to the legitimate connections, thus avoiding exceeding the detection threshold. When the host is otherwise idle, however, as long as the worm does not exceed the BDD thresholds, it is then free to make first-contact connections. Figure 2(c) shows the legitimate traffic with an adaptive worm overlaid. By scanning mostly when the host is in the middle of a quiescent period, the adaptive worm avoids having a scanning rate greater than the legitimate traffic, even at a higher scanning rate than the classic worm (with eight worm scans instead of five).

Preventing or limiting this adaptive behavior of worms would then help to reduce the achievable scan rate of a worm, and is the basis for QPD. Basically, if a host does not display quiescent periods as it typically does, and has been "active" for overly long, QPD then determines that the host is infected by a worm that is scanning the network.

QPD thus detects worms by measuring the duration of active periods during a training phase. An active period is defined as the duration of a period during which first-contact connections happen with *no more than* the specified quiescent period between them. QPD uses a series of different quiescent periods. For every quiescent period size, it measures the mean and standard deviation of all the active periods that are

separated by a quiescent duration of at least that length. These values are used to generate a threshold duration for active periods, which is the mean plus β times the standard deviation. β can be tweaked for different environments to fix the false positives at a specific value. If a host has an active period exceeding the threshold duration for any of the quiescent period, it is likely infected with a worm. For example, we can apply QPD to Figure 2(c) where the host is active all the time and does not exhibit any quiescent period at all to detect the presence of the worm.

D. Clustering

Existing behavior-based detection systems employ the same threshold for all hosts in a protected network. This is a poor choice because the hosts in a network show widely divergent behaviors. As more devices (*e.g.*, handheld devices) connect to the Internet, they also come with more divergent behaviors. Desktop computers used primarily for web surfing make connections in a different pattern than a department email server would, for example. If a desktop computer started making connections at the same rate as the email server, it is likely an anomalous event and something strange must have happened to that computer. But if the desktop computer has same thresholds applied to it that the email server does, its behavior would not appear to be anomalous because those thresholds must allow it as normal behavior to avoid constantly flagging the email server as infected.

In fact there has been substantial research effort in clustering entities based on their expressed characteristics in both the realm of statistics and computer science. We have leveraged this body of work and applied existing tools to automatically categorize the hosts in a network such that different thresholds can be applied to different groups of hosts. We examined a range of clustering techniques, behavior characteristics to cluster against, and number of clusters to create. We have found that using *k*-means clustering to separate the hosts into groups allows us to improve overall performance. In our current design we cluster based on a single feature of the hosts, the number of destinations contacted during a training period.

E. The SWORD Worm Detector

We have combined the above principles into a new worm detector, *i.e.*, SWORD. It uses the BDD and QPD detectors outlined above, and declares a host to be infected with a worm when either BDD or QPD raises an alarm. SWORD observes legitimate network activity for a period of time to cluster hosts into groups and generate thresholds for each cluster.

The co-existence of BDD and QPD makes it extremely hard for a worm to avoid being caught. If a worm wants to escape BDD but still makes new connections, it cannot shorten the duration of a burst of any size; it will then have to lengthen active periods, but doing so will get it caught by QPD. On the other hand, if a worm wishes to escape QPD while still making new contacts, it then has to ensure the quiescent periods; it will then have to insert its connections into active periods, which however will cause certain connection bursts to have a shorter duration than permitted, thus triggering the alarm from BDD. Therefore, this combined, collective detection of SWORD captures the fundamental behavior of worm detection, preventing a worm from quickly spreading to many destinations.

IV. Experiment Methodology

We now describe how we evaluate a worm detector, especially its performance against evasive worms. Our objective is that we can use realistic trace to evaluate the performance of different worm detectors against all kinds of worms with various different parameters.

A. Procedure Overview

We developed a custom testing framework that allows us to easily plug in a behavior-based detector, set up an evaluation environment with both background traffic and worm traffic, and run the detector to measure its performance in detecting the existence of the worm traffic. We implemented SWORD in this framework, as well as all detectors described in Section II, *i.e.*, DSC, TRW, RBS, MRW, PGD, and TRWRBS.

For each worm detector, including SWORD, we first run it against classic worms, which are not aware of the detector against it at all. We measure its accuracy (*i.e.*, false positives and false negatives) and detection latency at different scanning rates. We then run every detector against evasive worms which are designed specifically to evade the detector, and see how resilient a detector is against evasive worms. We describe evasive worms in Section IV-B, and detail the metrics and parameters for measuring against both classic and evasive worms in Section IV-C.

We run each detector in four distinct environments: *campus*, *enterprise*, *department*, and *wireless*. Every environment includes background traffic from a real source, and worm traffic with a variety of worm scanning strategies generated using the GLOWS worm simulator [17] that is tailored to that environment. Note we ensure the worm traffic is unbiased toward SWORD. As the traffic behavior in each environment is different, such as the durations of traffic bursts and durations of active periods, SWORD may demonstrate different performance under different environments. We detail them in Section IV-D.

To evaluate SWORD's performance against other detectors, we perform the same set of experiments using the same network traces and worm simulations, allowing us to fairly compare different detectors and objectively assess their performance.

B. Evasive Worms

For a worm to evade detection, it must know the underlying details of the detector being used, then leverage its capabilities to adjust its behavior in order to avoid triggering detection. We define several terms to refer to capabilities of evasive worms. A worm with no knowledge of the legitimate network traffic on an infected host is said to be **blind**, whereas if it can observe the traffic it is **perceptive**. A worm that does not know the parameters of the detector deployed against it is described as **speculative**, whereas one that knows the actual deployed parameters is said to be **informed**. We consider all permutations of these capabilities.

For each worm detector we evaluate, we implemented a different type of evasive worm against it. Specifically, the evasive worm against DSC adds a delay between infecting a host and the beginning of scanning from that host to avoid any

causality connection; the evasive worm against TRW contacts a list of known hosts to avoid connection failures; the evasive worms against MRW and PGD both scan at the maximum sustained rate that will not be detected; and the evasive worm against TRWRBS is a combination of the evasive worms against TRW and RBS. (We do not consider evasive worms against RBS because RBS does not even perform well against classic worms, as shown in Section V-A.) For details on how an evasive worm with regard to a specific worm detector functions, readers can refer to [18].

C. Metrics and Parameters

For each detector in each environment, we first run it against benign traffic with no injected worm activity. The **false positive rate** is the number of hosts misidentified as infected per hour. We then run 16 experiments for every permutation of the worm parameters (*e.g.*, we run 16 experiments to measure a random-scanning worm at every scanning rate). Each experiment consists of running the detector for 10 minutes of the experiment trace to warm up the connection histories, then injecting the simulated worm traffic into the trace, and running until either an hour has elapsed or the worm is detected. Each of the 16 experiments that we run for a given set of worm parameters has a different host in the protected network being infected first and uses a different random seed. The **false negative rate** is then the percentage of experiments where the worm is not detected, and the **detection latency** is the mean number of worm connections that have left the network at detection time.

For each evasive worm we vary a parameter ζ between zero and one, controlling the aggressiveness of the scanning. A value of one means that the worm will modulate its own scanning traffic to bring the detector exactly to its threshold. A value of zero would cause the worm to modulate its traffic so that it presents no signature of the worm (if possible), which of course might mean that the worm would not scan at all. We call this parameter **load factor** because the worm traffic is additional load against the detector threshold beyond whatever legitimate traffic originates from the host.

We run each evasive worm once in each environment for 1 hour for each of 16 different randomly selected first infected hosts, for 10 different values of ζ in the range [0.1–1.0], and with an upper bound of scanning rate at 10 scans per second.

We use three metrics to evaluate the success of an evasive worm. As we vary the load factor, we measure the worm's ability to evade detection: its **evasion rate**. This represents the percentage of experiments where the evasive worm is *not* detected by the worm detector in place. Clearly, the evasion rate of the worm is equivalent to the false negative rate of the corresponding worm detector.

The second metric is the **effective scanning rate**. This is the average number of worm scans per second the evasive worm is able to make during the one hour experiment for a given environment and value of ζ. The higher the value of ζ, the faster the evasive worm will scan, thus increasing its effective scan rate (but also reducing the evasion rate).

An evasive worm author's goal is to scan as quickly as possible while maintaining a high chance at evasion. As the load factor of a worm increases, its scanning rate also increases, but its evasion rate decreases. By choosing a minimum evasion rate, we then can find the **maximum effective scanning rate**. For this experiment we choose the minimum evasion rate to be 0.90, meaning the worm can survive if the false negative rate of detection is 90% or higher. The maximum effective scanning rate is the ultimate determination of a worm detector's effectiveness. The lower the maximum effective scanning rate allowed, the more effective a detector is. This single metric is the best metric for comparing detectors, as it reveals the damage that a worm can cause without being detected.

D. Evaluation Environments

The *campus* environment is built from a trace collected at the border of Auckland University [19]. It contains over a month of traffic from the entire university with two /16 and several /24 networks. We randomly select 200 hosts and construct an environment using traffic to and from those hosts, where the training and experiment segments each contain approximately 25,000 connections.

The *enterprise* environment is built from a trace collected at LBNL [20]. Heavy scanners were removed from the trace before it was released. It has 139 active hosts and the training and experiment segments each also contain roughly 25,000 connections.

The *wireless* and *department* environments are built from traces collected at the University of Massachusetts [21]. The department environment is built from a trace capturing all traffic to and from the wired computers in the CS department. It has 92 active hosts and approximately 30,000 connections in each training or experiment segment. The wireless environment comes from a trace capturing all wireless network traffic from the university. It has 313 active hosts and approximately 120,000 connections in each segment.

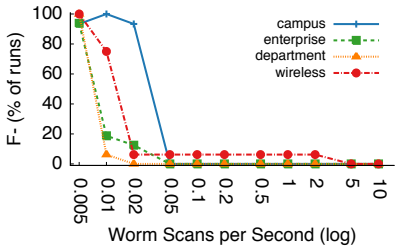
V. Evaluating the SWORD Detector

Having described SWORD in Section III, we now evaluate its ability to detect worms and compare it with the detectors we described in Section II. Not only do we consider SWORD's performance against classic worms, we also measure its performance against evasive worms that attempt to avoid exhibiting the behaviors that SWORD relies on. We measure evasive worms' effective scanning rate, evasion rate, and maximum effective scanning rate against SWORD and show that SWORD is more effective at limiting an evasive worm's spread than the state-of-the-art detectors.

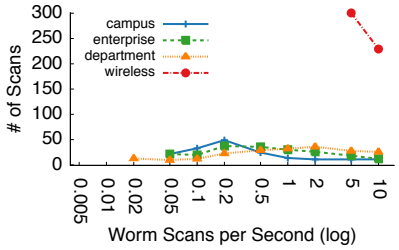
A. Performance of SWORD vs. Classic Worms

We now describe the performance of the SWORD detector against classic worms in terms of its accuracy and detection latency. We report the results using *random-scanning worms* as the classic worms. Our experiments show that SWORD has similar performance in detecting other classic worms of different scanning types, such as the local-preference worms, topological-scanning worms, or hitlist worms, whereas other detectors demonstrate either similar performance in detecting other classic worms, or worse.

Figure 3(a) shows the false negative rate that SWORD achieved against a classic worm. The worm was detected at a scanning rate of 0.05 connections per second in every scenario except for a single host in the wireless environment.



(a) False Negatives



(b) Detection Latency

Fig. 3: **False negative and detection latency of SWORD** when running against classic worms infecting randomly selected hosts.

To make the direct head-to-head comparison between SWORD and other detectors easier, Figure 4 further plots SWORD and the other detectors all on the same graph. All these detectors are adjusted to have the same false positive rate (two falsely identified hosts per hour). In the campus environment (Figure 4(a)), we can see that the TRW detector is able to detect some worms at slightly slower scan rates than SWORD. It is the only detector to do so, however, and does not detect 100% of the infections at any scan rate slower than SWORD does. The enterprise environment shows similar results (Figure 4(b)), again with TRW showing slightly better sensitivity and this time PGD just barely beating SWORD at 0.02 scans per second. The other two environments, however, show SWORD with the best sensitivity, detecting worm infections at slower scanning rates than any other detector (Figure 4(c) and Figure 4(d)).

Figure 3(b) shows the detection latency of SWORD. In the campus, enterprise, and department environments, the average detection latency is under 40 scans for all worm scanning rates but one, where the average detection latency is under 50 scans. We do not see much of the latency performance in the wireless environment, because we only plot detection latency for combinations of environment and scanning rate where the worm was detected in 100% of the experiments. However, if we were to relax our restriction and show the detection latency for those scenarios at each scanning rate where the worm was detected in the wireless environment, we would see that the latency is under 67 for all scan rates under 0.2, and under 327 for all worm scan rates. In comparing SWORD and the other detectors in terms of their detection latency, it is hard to plot a graph similar to Figure 4 with good legibility because of the wide ranging latency values. Instead, we present a simplified view, with the average taken across all scanning rates where the worm was detected for each detector and environment. These results are presented in Table I. (If a detector is faster than SWORD its latency is highlighted.) Clearly, in most cases SWORD is faster than other detectors.

TABLE I: **Average detection latency for all detectors.**

Detector	Campus	Enterprise	Department	Wireless
SWORD	21.73	24.97	22.99	264.94
DSC	2.00	22.00	19.00	15.93
MRW	28.88	51.70	43.64	1014.16
PGD	93.80	28.11	25.81	621.75
RBS	17.36	4.25	26.44	349.53
TRW	4.23	11.13	24.75	49.93
TRWRBS	57.97	30.39	58.66	167.95

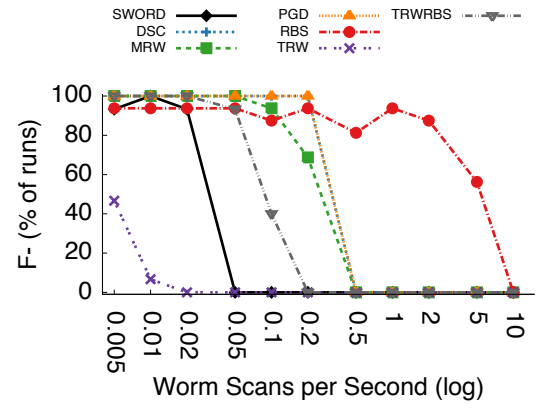
Overall, in detecting classic worms, the only two detectors that *sometimes* beat SWORD are PGD (in one environment only) and TRW (in two environments). SWORD has a lower average detection latency than PGD in all environments here, including a latency of less than half in the wireless environment. The TRW detector has a lower detection latency in three of the four environments. These tests make the TRW detector appear to be a better detector than SWORD. However, a clever worm can evade the TRW detector by employing known neighbors to befuddle the detector. In the next section we show that the SWORD detector is dramatically better once evasive worms are taken into account. Against all other detectors, SWORD has either better sensitivity or detection latency, and in many cases both.

B. *Performance of SWORD vs. Evasive Worms*

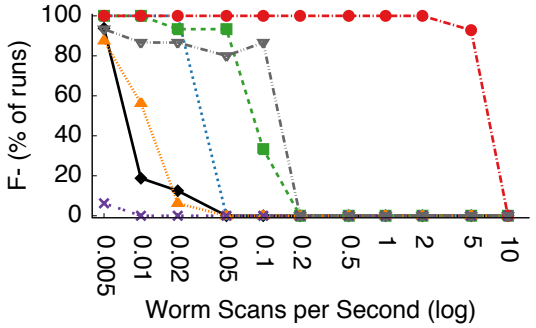
A good worm detector must be effective not only against classic worms, but also against worms that are actively trying to evade detection. We now evaluate SWORD against evasive worms. We test SWORD against *blind/perceptive* and *speculative/informed* worms and measure the worms' effective scan rate and evasion rate. We then measure the maximum effective scan rate achieved with a less than 10% chance of detection.

1) Evading the SWORD Detector: An evasive worm against SWORD must ensure that it has sufficient quiescent periods to evade QPD, while also limiting its bursts of connections to avoid triggering BDD. The combination of these two mechanisms puts significant constraints on the ability of the worm to scan. More specifically, the worm runs internal versions of both the QPD and BDD detectors. For every scan to initiate, it first checks to see whether the scan will violate any of the QPD constraints. If it will, the worm waits long enough to end the current active period for the QPD constraint in question. After eliminating QPD as a constraint, it checks the BDD durations to ensure that the BDD detector will not be triggered either. Recall if the worm is perceptive rather than blind, it knows the legitimate traffic; and if it is informed rather than speculative, it know the parameters of SWORD.

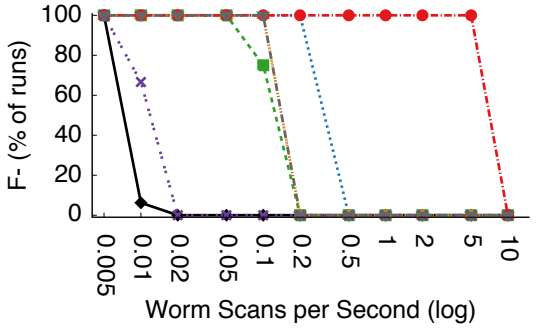
2) Effective Scan Rate and Evasion Rate of Evasive Worms against SWORD: The blind speculative version of the worm cannot achieve an effective scan rate of greater than 0.03 scans per second in any scenario (Figure 5(a)). In the department environment, when the load factor is 1, a scan rate even this low still gives an evasion rate of 0% (Figure 6(a)). The perceptive speculative version of the worm does not improve the effective rate at all (Figure 5(c)), but does improve the evasion rate in all but the department environment (Figure 6(c)). The informed versions of the worm do much better in the wireless environment, with the worm able to achieve



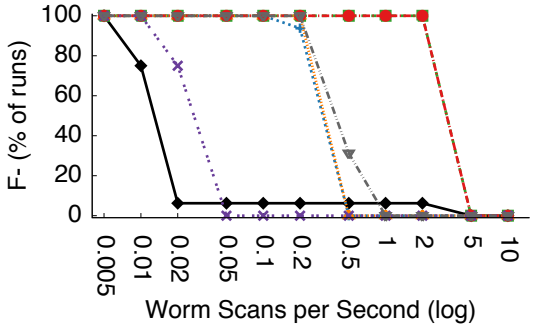
(a) Campus



(b) Enterprise

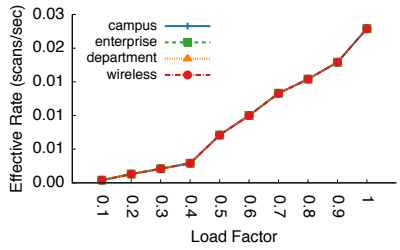


(c) Department

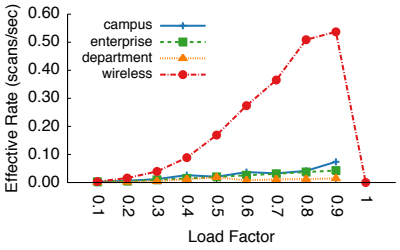


(d) Wireless

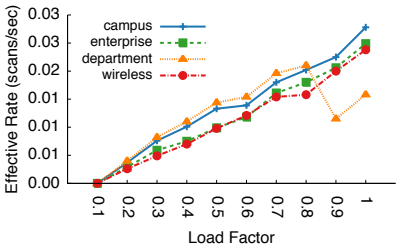
Fig. 4: **False negatives for SWORD and other detectors** when running against classic worms infecting randomly selected hosts. All these detectors are adjusted to have the same false positive rate (two falsely identified hosts per hour), so they can be compared against each other head-to-head.



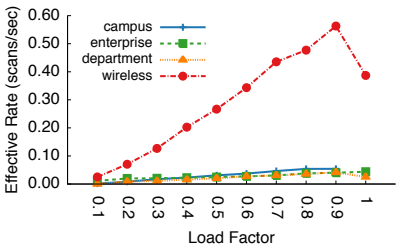
(a) Blind Speculative Worm



(b) Blind Informed Worm



(c) Perceptive Speculative Worm



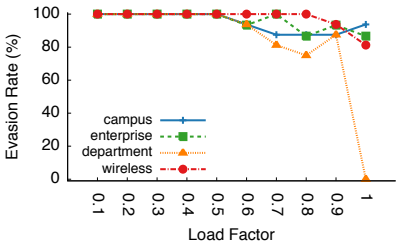
(d) Perceptive Informed Worm

Fig. 5: **Effective scanning rate of evasive worms vs. SWORD** as a function of Load Factor.

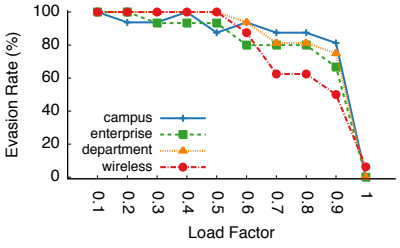
an effective rate nearly 10x greater than the speculative worm was able to (Figure 5(b) and Figure 5(d)). Overall, SWORD works effectively against evasive worms and can limit both the effective scanning rate and evasion rate of evasive worms.

3) Maximum Effective Scan Rate of Evasive Worms Against SWORD and Existing Detectors: The best evaluation of the detector is the maximum effective rate achieved by the evasive worm while running less than a small chance (we use 10%) of being detected. In Figure 7, we plot the maximum effective rate achieved by the evasive worms against respective detectors.

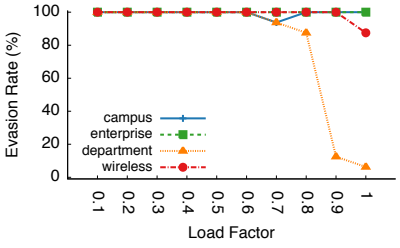
For the campus environment (Figure 7(a)), the benefits of the SWORD detector are very pronounced, beating all other detectors by at least a factor of 2. In this environment, no other detector came close to limiting the maximum effective rate of the evasive worms as well as SWORD did.



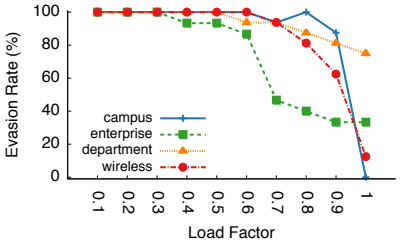
(a) Blind Speculative Worm



(b) Blind Informed Worm

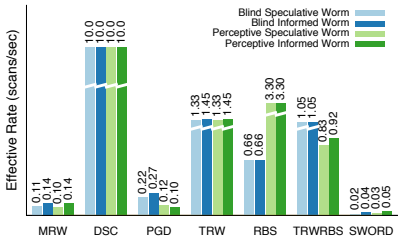


(c) Perceptive Speculative Worm

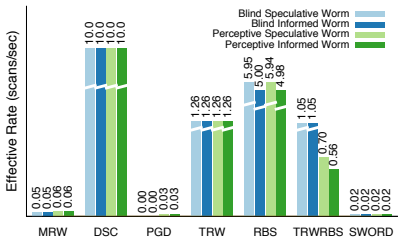


(d) Perceptive Informed Worm

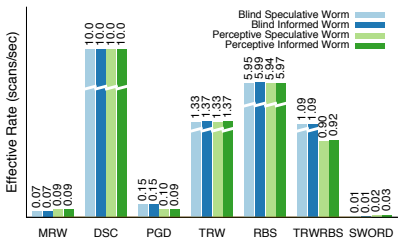
Fig. 6: **Evasion rate of evasive worms vs. SWORD** as a function of Load Factor



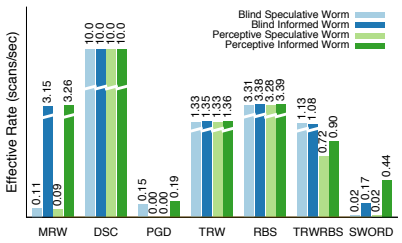
(a) Campus



(b) Enterprise



(c) Department



(d) Wireless

Fig. 7: **Maximum effective rate of evasive worms against their detectors.**

In the enterprise environment (Figure 7(b)), the maximum effective rate allowed by the SWORD detector to any variety of evasive worm was 0.02 scans per second. The PGD detector is the only one competitive against SWORD, while the other detectors all perform significantly worse. PGD detected the blind evasive worm varieties in every scenario in this environment because of the window that had a threshold that was only one scan over the legitimate traffic for this period. However, the perceptive variant of the worm was able to achieve a maximum effective rate of 50% greater against the PGD detector than against the SWORD detector.

Figure 7(c) shows similar results for the department environment. SWORD outperforms all other detectors by at least a factor of three for all evasive worm varieties.

Finally, Figure 7(d) shows that again, the PGD detector was able to outperform SWORD in some scenarios. But even here, SWORD outperforms PGD by a factor of 7 against blind speculative worms.

C. Summary

In this section we have shown that the SWORD detector significantly outperforms all other detectors. The TRW detector does perform slightly better than SWORD against classic worms, but against evasive worms it was outperformed by a factor of 60 in most environments against most evasive worm types. TRW's best performance against evasive worms was coming within a factor of 3 against a perceptive informed worm in the wireless environment. This significant superiority against evasive worms offsets any minor advantage TRW had

over SWORD against classic worms. The PGD detector does outperform SWORD in 5 of the 16 evasive worm scenarios (four evasive worm types by four environments), but is dominated in the remaining 11 scenarios. It is also outperformed by SWORD against classic worms. The only other detector to come close is the MRW detector, which is consistently outperformed by SWORD against classic worms. MRW's best performance against evasive worms is to get within a factor of two of the SWORD detector. It is soundly beaten in every scenario. None of the other detectors present even an appreciable level of competition.

VI. Limitations and Open Issues

The evasive worm against every detector could be further improved. For example, the current design of the evasive worm against TRW assumes that the worm can always find a list of known hosts to contact, but sometimes it may be impossible. Also, the experimentations above assume that the training is reliable, but if the background traffic is infected by a worm, it can introduce noise to detection results. Lastly, it will also be interesting to more closely analyze why every detector, including SWORD, tends to perform differently under a different environment.

VII. Conclusions

We identify two principles that an effective worm detection solution must follow: (1) Worm propagation and worm detection are in an arms race, and a detector must consider potential countermeasures from worm authors; and (2) Behavior-based worm detection must focus on the fundamental behavior of worm propagation that worms cannot avoid. Although there have been already many existing worm detectors, they are all inadequate in following these two principles.

In this paper, we revisited behavior-based worm detection. We identified that the fundamental behavior of worm propagation is that of connecting to new destinations, and designed a new detector SWORD that encompasses two complementary modules that can detect violations that a worm will cause in connecting to new destinations. With one module monitoring burst duration and the other ensuring quiescent periods, we show SWORD is extremely hard for a worm to evade. As demonstrated in our evaluation, the SWORD detector significantly outperforms all other detectors. SWORD is not only as competitive as the best detector for detecting classic worms, but is also highly resilient against evasive worms.

References

[1] S. Staniford, V. Paxson, and N. Weaver, "How to 0wn the Internet in your spare time," in *Proceedings of the USENIX Security Symposium*. Berkeley, CA: USENIX, 2002, pp. 149–167.

[2] Z. Liang and R. Sekar, "Fast and automated generation of attack signatures: A basis for building self-protecting servers," in *Proceedings of the Conference on Computer and Communications Security*. New York, NY: ACM Press, 2005.

[3] J. R. Crandall, Z. Su, S. F. Wu, and F. T. Chong, "On deriving unknown vulnerabilities from zero-day polymorphic and metamorphic worm exploits," in *Proceedings of the Conference on Computer and Communications Security*. New York, NY: ACM Press, 2005.

[4] J. Newsome and D. Song, "Dynamic taint analysis for automatic detection, analysis, and signature generation of exploits on commodity software," in *Proceedings of the Network and Distributed System Security Symposium*. Reston, VA: The Internet Society, February 2005.

[5] J. Tucek, J. Newsome, S. Lu, C. Huang, S. Xanthos, D. Brumley, Y. Zhou, and D. Song, "Sweeper: A lightweight end-to-end system for defending against fast worms," in *Proceedings of the EuroSys Conference*, 2007.

[6] H.-A. Kim and B. Karp, "Autograph: Toward automated, distributed worm signature detection," in *Proceedings of the USENIX Security Symposium*. Berkeley, CA: USENIX, August 2004, pp. 271–286.

[7] S. Singh, C. Estan, G. Varghese, and S. Savage, "Automated worm fingerprinting," in *Proceedings of the Symposium on Operating System Design and Implementation*. Berkeley, CA: USENIX, 2004, pp. 45–60.

[8] K. Wang, G. Cretu, and S. J. Stolfo, "Anomalous payload-based worm detection and signature generation," in *Proceedings of the Symposium on Recent Advances in Intrusion Detection*. Berlin, Heidelberg: Springer-Verlag, 2005.

[9] K. Wang, J. J. Parekh, and S. J. Stolfo, "Anagram: A content anomaly detector resistant to mimicry attack," in *Proceedings of the Symposium on Recent Advances in Intrusion Detection*. Berlin, Heidelberg: Springer-Verlag, 2006.

[10] Z. Li, L. Wang, Y. Chen, and Z. Fu, "Network-based and attack-resilient length signature generation for zero-day polymorphic worms," in *Proceedings of the IEEE International Conference on Network Protocols*. Washington, DC: IEEE Computer Society, October 2007, pp. 164–173.

[11] S. Stafford and J. Li, "Behavior-based worm detectors compared," in *13th International Symposium on Recent Advances in Intrusion Detection (RAID)*, Ottawa, Canada, September 2010, p. 20 pages.

[12] G. Gu, M. Sharif, X. Qin, D. Dagon, W. Lee, and G. Riley, "Worm detection, early warning and response based on local victim information," in *Proceedings of the Annual Computer Security Applications Conference*. Washington, DC: IEEE Computer Society, 2004.

[13] S. E. Schechter, J. Jung, and A. W. Berger, "Fast detection of scanning worm infections," in *Proceedings of the Symposium on Recent Advances in Intrusion Detection*. Berlin, Heidelberg: Springer-Verlag, 2004.

[14] V. Sekar, Y. Xie, M. K. Reiter, and H. Zhang, "A multi-resolution approach for worm detection and containment," in *Proceedings of the International Conference on Dependable Systems and Networks*. Washington, DC: IEEE Computer Society, 2006.

[15] M. P. Collins and M. K. Reiter, "Hit-list worm detection and bot identification in large networks using protocol graphs," in *Proceedings of the Symposium on Recent Advances in Intrusion Detection*. Berlin, Heidelberg: Springer-Verlag, September 2007, pp. 276–295.

[16] J. Jung, R. Milito, and V. Paxson, "On the adaptive real-time detection of fast-propagating network worms," in *Proceedings of the Conference on Detection of Intrusions and Malware and Vulnerability Assessment*. Berlin, Heidelberg: Springer-Verlag, July 2007, pp. 175–192.

[17] S. Stafford, J. Li, T. Ehrenkranz, and P. Knickerbocker, "GLOWS: A high fidelity worm simulator," University of Oregon, Tech. Rep. CIS-TR-2006-11, 2006.

[18] S. Stafford, "Behavior-based worm detection," Ph.D. dissertation, University of Oregon, Department of Computer and Information Science, March 2012, advisor: Jun Li.

[19] WAND Network Research Group, "WAND WITS: Auckland-IV trace data," http://wand.cs.waikato.ac.nz/wand/wits/auck/4/, April 2001.

[20] Lawrence Berkely National Laboratory, "LBNL/ICSI enterprise tracing project," http://www.icir.org/enterprise-tracing/, 2005.

[21] University of Massachusetts Amherst, "Umass trace repository," http://traces.cs.umass.edu/, 2008. [Online]. Available: http://traces.cs.umass.edu/