Detecting Zero-day Self-propagating Internet Worms Based on their Fundamental Behavior

Shad Stafford — staffors@cs.uoregon.edu
Toby Ehrenkranz — tehrenkr@cs.uoregon.edu

NetSec Lab - University of Oregon
Principle Investigator: Jun Li (lijun@cs.uoregon.edu)

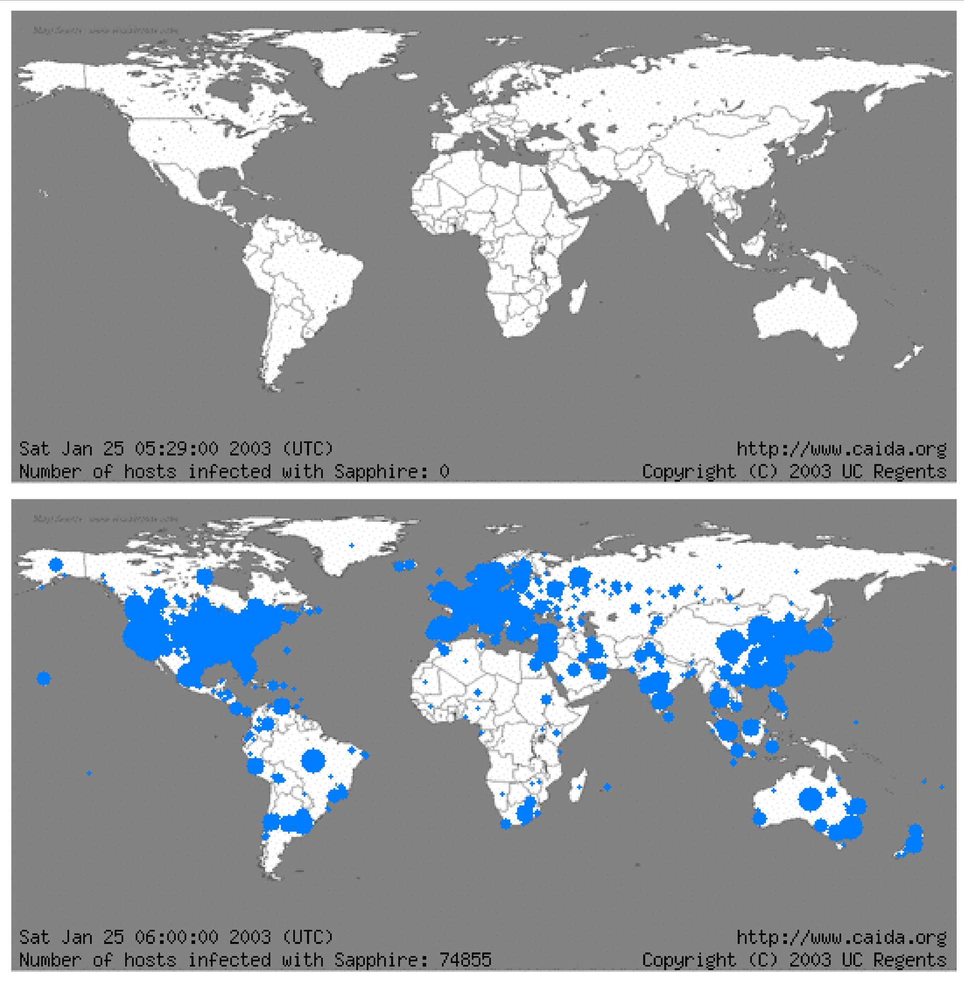
The Worm Menace

• The Internet is now a critical infrastructure and is at risk of shutdown due to worm activity. The Code-Red and Sapphire/Slammer worms are estimated to have cost $3 billion dollars in damages and lost productivity.

• Sapphire/Slammer achieved significant penetration in less than 30 minutes (see figures at right), but current worm countermeasures require the manual creation of byte-stream signatures, a process that can take hours or days.

• To counter this threat, a fast, automated worm defense system is required.

• We present SWORD, our worm detection system which uses host-level behavior to automatically detect current and future worms. It will form the basis for an automated response system that is fast enough to save the Internet.



How to Catch a Worm

Rather than identifying worms based on byte-stream signatures, we focus on some intrinsic behaviors of worms:

• *Self-replication/Self-similarity:* Worms generate many causally-related self-similar connections as they propagate from host to host. Worm traffic will show this characteristic while normal traffic may or may not.

• *Destination Address Distribution:* Worms attempt to connect to a large number of different hosts in a pattern that is quite different than what is seen in normal traffic.

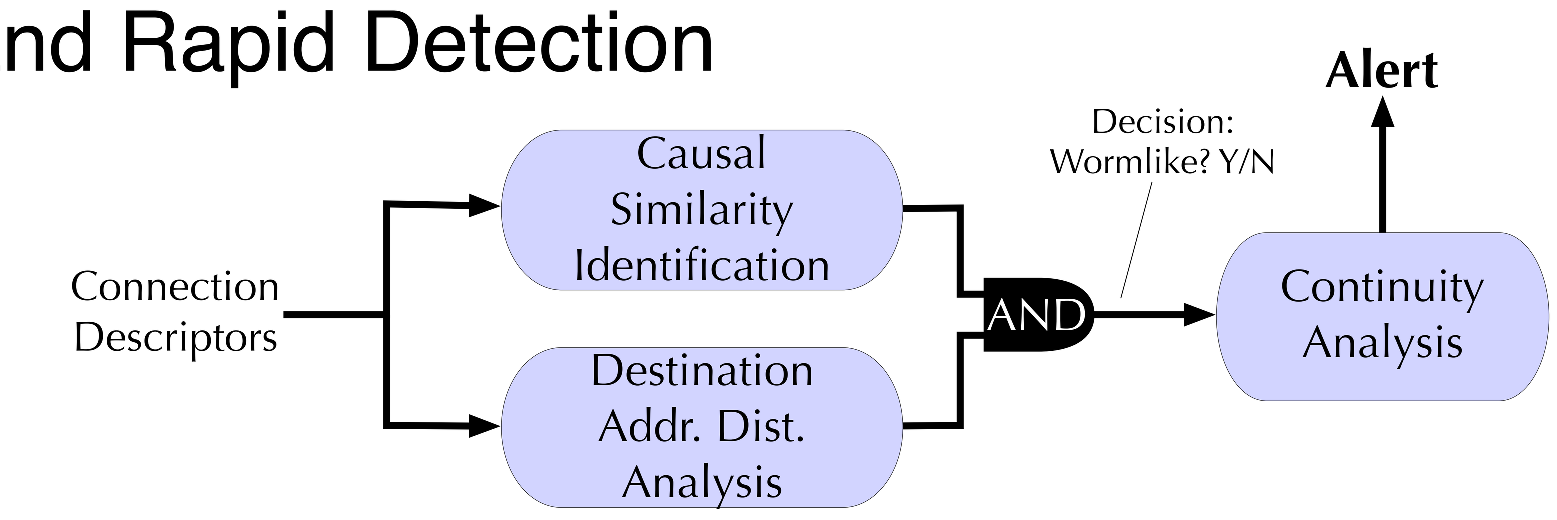
• *Continuity Analysis:* Worms are persistent in their scanning attempts. Although legitimate traffic may occasionally show wormlike behavior, only true worms will do so consistently.

SWORD: Self-propagating Worm Observation and Rapid Detection

• It is prohibitively expensive to monitor all connections on a network. Instead we place a monitor at the gateway to the network and watch traffic there.

• To further simplify the data, we coalesce packets into Connection Descriptors and do our analysis at the connection level. This avoids pitfalls with polymorphic or encrypted payloads.

• Our monitor applies two algorithms to each connection and when they both agree that the connection is wormlike, it is tallied in a sliding window. When the wormlike connection tally exceeds a given threshold, an alert is raised.

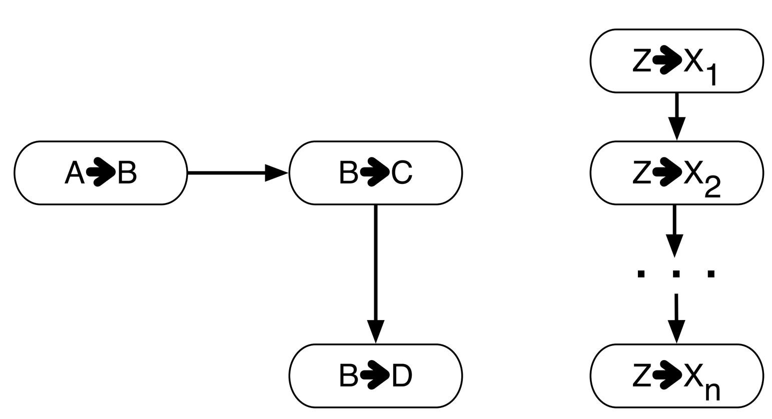


Causal Similarity Identification

• Adds each connection to a *causal connection graph* representing all the possible causes of a connection using a mechanism similar to Lamport's *happened-before* relationship.

• For each new connection, compare it to its ancestors in the causal connection graph. If enough are similar, this connection is considered wormlike.

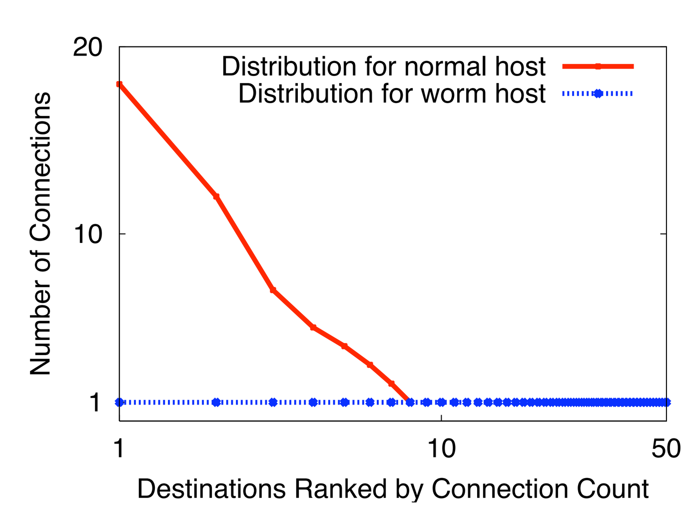
• The two examples below show different subgraphs that might result in identifying a connection as wormlike. The first shows host B becoming infected and beginning to scan. The second shows host Z scanning without a visible infection.



Destination Distribution Analysis

• The normal connection pattern for a host is to connect to a limited set of addresses such that the connection history maps to a zipf-like distribution. Worms on the other hand, typically make connections to a large set of target addresses.

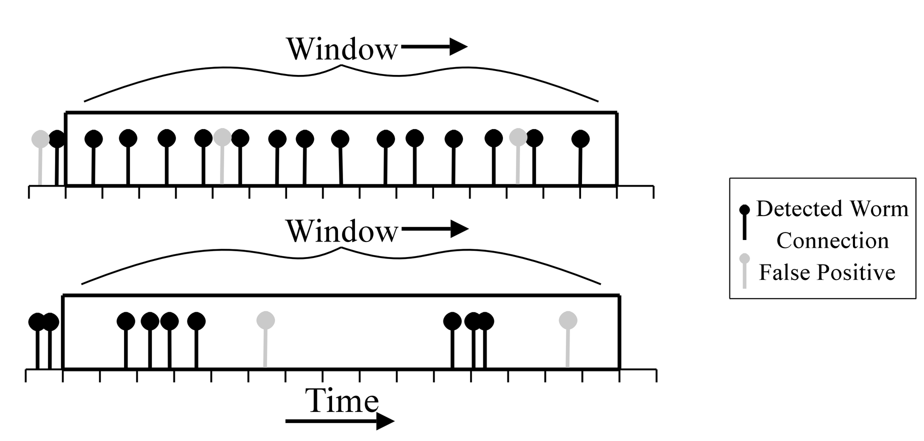
• This algorithm examines the connection history of each host every time it makes a new connection. When a new connection causes the distribution to trend sufficiently away from matching a zipf-like pattern, the connection in question is considered wormlike.



Continuity Analysis

• Worm connections are consistently identified as wormlike by the detection algorithms, but legitimate connections may also be identified as wormlike. These connection-level false positives must not cause the detector to mistakenly raise an alert when no worm is present.

• SWORD employs a sliding-window to filter out the noise of connection-level false positives. Only when the number of wormlike connections within a window exceeds a specified threshold is a worm alert raised.



Experimental Results

• To evaluate SWORD, we ran it against a trace consisting of real traffic recorded at a gateway router at Auckland University combined with worm traffic simulated on the Auckland network topology.

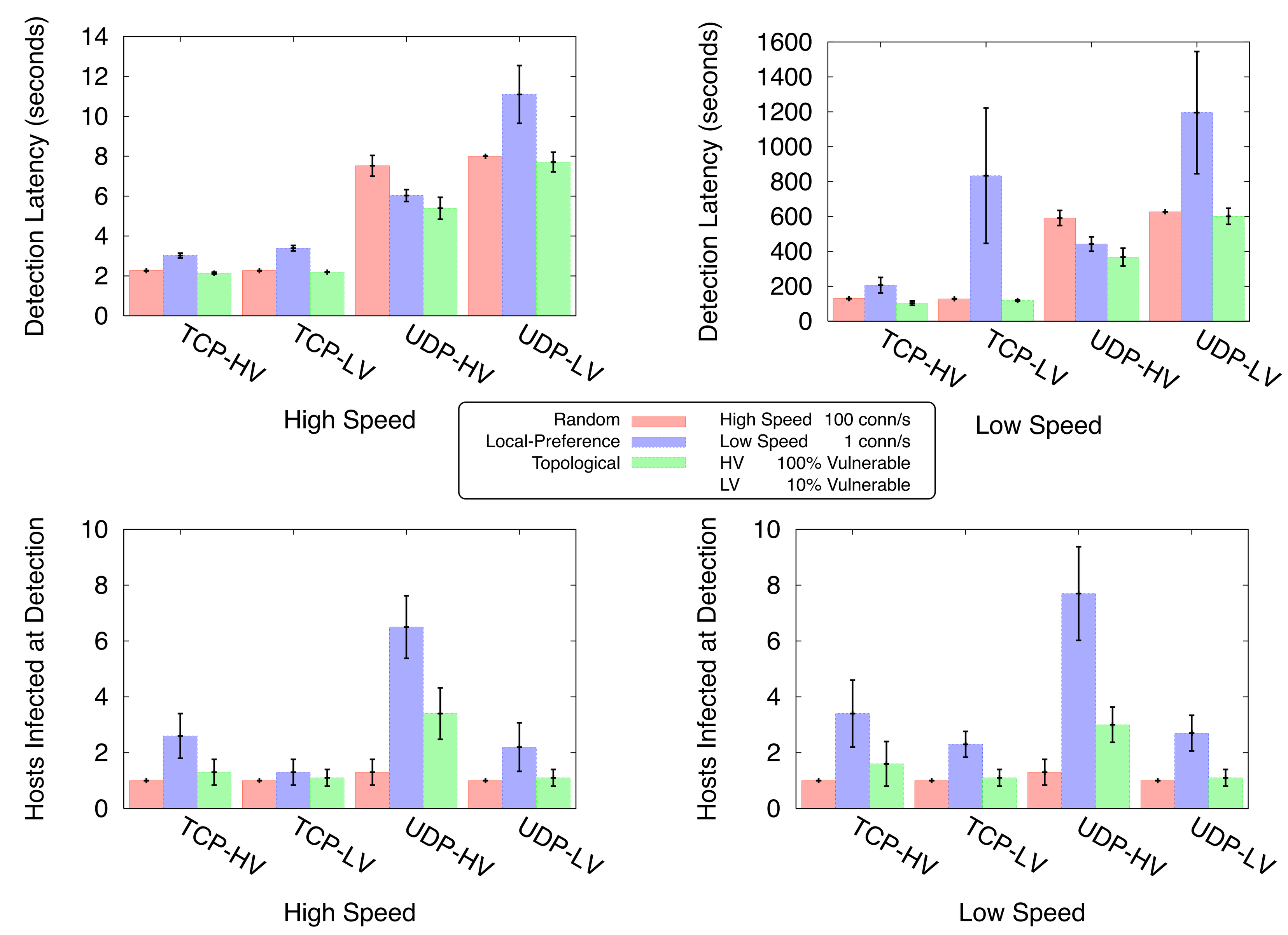
• We tested against *random, local preference,* and *topological* scanning worms. To give broad coverage of the threat spectrum we ran multiple simulations with varying vulnerability levels and worm scan rates. Each experiment was run 10 times with different random seeds.

• Our metrics are *accuracy* and *latency*. Accuracy is the percent of experiments where we correctly identified the presence or absence of a worm. Latency is measured as both the *time* between first infection and detection, and the *number of hosts* infected at detection.

• In our experiments SWORD was 100% accurate. It always detected the presence of a worm and never reported a worm when one was not present.

• The graphs to the right show the detection latency of SWORD. In all cases the worm was detected with fewer than 10 out of 5000 hosts infected. High-speed worms were detected within 12 seconds and low-speed worms were detected within 20 minutes

• The overhead required to run SWORD is entirely manageable. Our offline evaluations were performed on a 1.73GHz Pentium M Laptop with 512 MB of RAM which was able to process a 24 hour trace containing 3 million connections in under one hour.



Supported by a grant from the Intel Corporation