Impeding worm epidemics through destination address filtering

Hyogon Kim and Inhye Kang

The effectiveness of destination IP address filtering against fast worm epidemics is analyzed. It is shown that its impeding effect depends heavily on the deployment ratio. If wide deployment is only achieved, which we argue is not as impractical as one might think, it can fundamentally block so called fast epidemics.

Introduction: Today's worm epidemics entered a regime where their spread is so overwhelmingly fast that traditional human-intervened response is no longer adequate. For instance, the SQL Slammer (a.k.a. Sapphire) epidemic infected most of the vulnerable nodes in just 10 minutes [1], while substantial response came in 2 to 3 hours world-wide [2]. Thus the response only stopped side-effects, not the worm itself. From the epidemiological model [3], we can readily prove that the infection speed is a function of, most importantly, the scanning rate and the vulnerable population size. Namely, the logistic equation below dictates the dynamics of the epidemic:

$$x(t) = \frac{K}{1 + \left(\frac{K}{x_0} - 1 \right) e^{-rt}} \qquad (1)$$

where $x(t)$ is the number of infected nodes at time t, K is the total number of vulnerable nodes, and $x(t=0)=x_0$ is the number of worms at the outset. $r=K/T*v$ is the infection rate, where v is the scanning rate, and T is the scanned space size. Since worms typically scan the entire IP address space, $T=2^{32}$. Figure 1 shows the time to 90% infection as a function of K, given v=10,000/s (c.f. in SQL Slammer, the average was 4,000/s and the recorded maximum was 26,000/s [1]), and x_0=1 (*i.e.,* worst condition for the worm). In essence, for a fast-scanning worm exploiting any vulnerable software with a substantial installed base (e.g. >100,000), the peak epidemic is attainable in just a few tens of seconds. Currently, against such fast worms we have absolutely no impeding element installed in the Internet infrastructure. Although there is a proposal for a signature-based content filtering inside network [4], its practicability is dubious due to the complexity and its inherent limitation against encryption and polymorphism that is increasingly employed by modern worms and viruses (e.g. Loveletter [5]; There are even worm generator softwares on the Web that automatize polymorphic worm generation [6]).

Destination address filtering: Almost without exception, worms randomly scan for victims [1,7] (although so called "hit-list" worm [3] accumulates the list of vulnerable nodes before the outbreak, in no major epidemic so far the captured worm body contained one). For instance, Code Red II generates an address within the same /8 prefix with probability of 1/2, /16 prefix with 3/8, others with 1/8 [7]. SQL Slammer does not have such preference, and it generates a random address with equal probability. The randomly generated IP address of a possible victim in this way can be illegal in one of two ways – Martian [8] or currently unallocated (a.k.a. "tarpit") [9]. Collectively, these illegal destination addresses occupy 44% of the entire IP address space [10]. (Although these blocks are subject to future allocation, the

procedure is slow enough [11] to reflect on the address checker.) Therefore, more than 2 out of 5 infection attempts through random scanning result in the violation of the address usage, and we can leverage this property to detect and filter them on the packet level. Furthermore, a host can be made to "quarantine" itself if it transmits a Martian or unallocated destination address more than a preset number of times, say q, in a given observation interval. In the quarantine, the host launches a self-audit, eventually killing the worm within. We can set q fairly low (e.g. 5) since in reality, there are few innocent hosts that habitually transmit to illegal addresses [10]. We will assume that the destination address filtering includes the quarantine as well as the packet-level filtering.

Expected impact: Suppose d is the fraction of vulnerable nodes that employ the destination based filtering. The address filtering effectively reduces the vulnerable population base by a factor of (1-d). Thus the infection rate in Eq. (1) changes to:

$$r' \approx (1-d)r \qquad (2)$$

The approximation is because we allow q scans before we quarantine the perpetrator. Note that K is the number of networked servers on the Internet (whose open service a malicious code can exploit). Today, only Web servers and possibly some P2P "servers" could be on the order of millions. Even then, K/T should be much less than 0.001. With such small K/T, the probability of an infection in q scans is:

$$1-\left(1-\frac{K}{T}\right)^q \approx \frac{Kq}{T} \approx 0$$

so we ignore it in Eq. (2). On the other hand, the probability of a perpetrating node not caught in q random scans is approximately [10]:

$$p_n(N,q) = \sum_{i=0}^{q-1} \binom{N}{i} 2^{-N}$$

which is a fast decreasing function of N [10]. $N=vL$ is the number of scan packets in a given observation interval L. Since v is so high in fast epidemics, enough to cause denial-of-service [1], the "impunity" probability above is negligible. Now we show how much time this mechanism buys us at the outset in which to react to an epidemic. Figure 2 plots the time to 10% infection as a function of d for four different values of K, ranging from 1,000 to 1,000,000. We notice that the address filtering effect becomes visible only at high level of d. So it will not help much to slow down a fast epidemic with partial deployment. However, as d approaches 1, it can arbitrarily impede the epidemic even for large K. In particular, for d=1, $x(t)$ in Eq. (1) (subject to the modification of Eq. (2)) remains at 1, independent of t. So the whole issue is reduced down to whether sweeping deployment is feasible or not. We argue that it is feasible. When the operating system platforms in the past large epidemics are limited to a small number and only one is exploited at a time, its deployment at hosts can be as sweeping and swift as in everyday operating system patch. For instance, Microsoft Windows operating systems regularly perform such automatic update, and some Linux publishers also provide automatic patches on demand [12]. And the update can also deal with the changes of illegal address blocks due to future allocation [11]. Finally, as for source address filtering [13] and the filtering at IPv4 routers [8], a standard or a "Best Current Practice (BCP)" could also be established for the destination address filtering at hosts.

Conclusion: Although the effect is obscure with limited deployment, destination address filtering is one of the few fundamental cures we have against so called fast worm epidemics. With ubiquitous deployment, the epidemics from the type of worm

that blindly scans potential victims can be almost completely prevented. Automatic operating systems update channels available today could be exploited to facilitate wide deployment.

Acknowledgements: This work was supported by Korea University Grant.

References

[1] CAIDA, "Analysis of the Sapphire Worm," http://www.caida.org/analysis/security/sapphire/, Jan. 2003.

[2] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "The spread of the Sapphire/Slammer worm," a NANOG presentation, http://www.nanog.org/mtg-0302/ppt/worm.pdf, March 2002.

[3] Stuart Staniford, Vern Paxson, and Nicholas Weaver, "How to Own the Internet in Your Spare Time," 11th USENIX Security Symposium, June 2002.

[4] D. Moore, C. Shannon, G. Voelker, and S. Savage, "Internet Quarantine: Requirements for Containing Self-Propagating Code,", IEEE Infocom, March 2003.

[5] S. Taylor, "The Complexities of Viral VB Scripts," European Institute for Computer Antivirus Research (EICAR) International Conference, 2001.

[6] VBS Worm Generator, http://vx.netlux.org/vx.php?id=tv07.

[7] CAIDA, "CAIDA analysis of Code Red," http://www.caida.org/analysis/security/code-red/.

[8] F. Baker, Requirements for IPv4 routers, *RFC 1812.*

[9] IANA, Internet Protocol Version 4 Address Space, http://www.iana.org/assignments/ipv4-address-space.

[10] H. Kim and I. Kang, “On the Effectiveness of Martian Address Filtering and its Extensions,” IEEE Globecom, Dec. 2003.

[11] A. Broido, E. Nemeth and K. C. Claffy, “Internet Expansion, Refinement, and Churn,” a NANOG presentation, Feb. 2002.

[12] Red Hat Linux, http:// http://sources.redhat.com/.

[13] P. Ferguson and D. Senie, “Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing,” RFC 2827.

Authors’ affiliations

H. Kim (Department of Computer Science and Engineering, Korea University, Anam-dong, Seongbug-gu, Seoul 136-701, Korea)

I. Kang (Department of Mechanical Information Engineering, University of Seoul, Jeonnong-dong, Dongdaemun-gu, Seoul 103-743, Korea)

Figure captions:

Fig. 1 Speed of epidemic for v=10,000

Fig. 2 Timescale of initial outbreak as a function of d

Figure 1

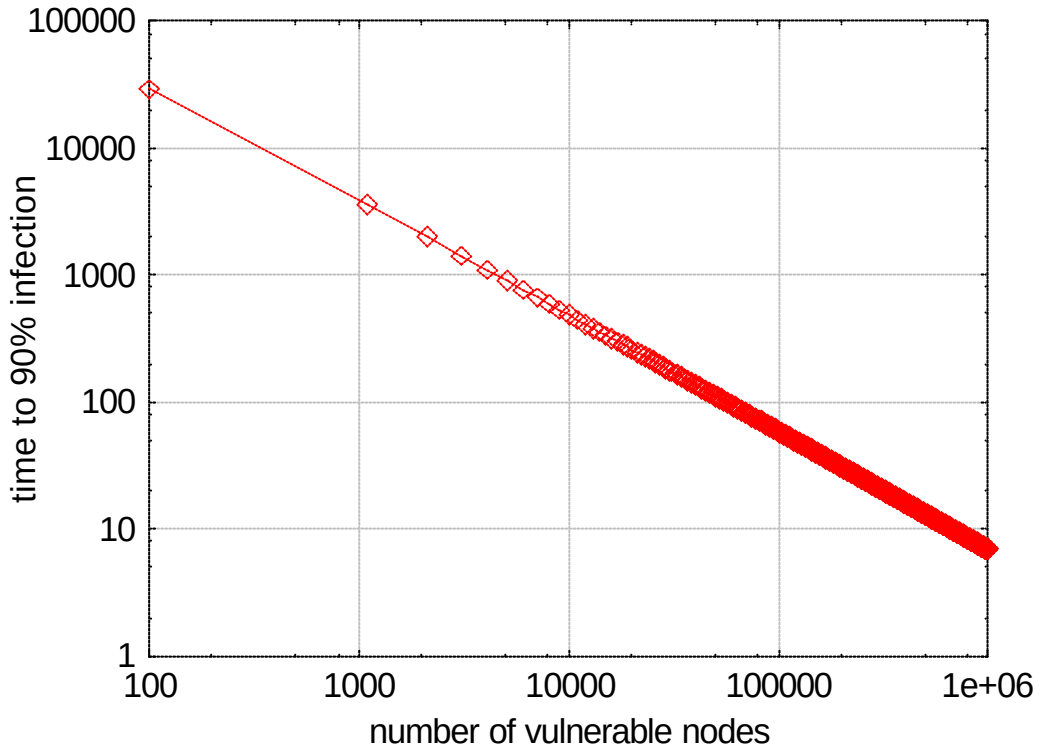
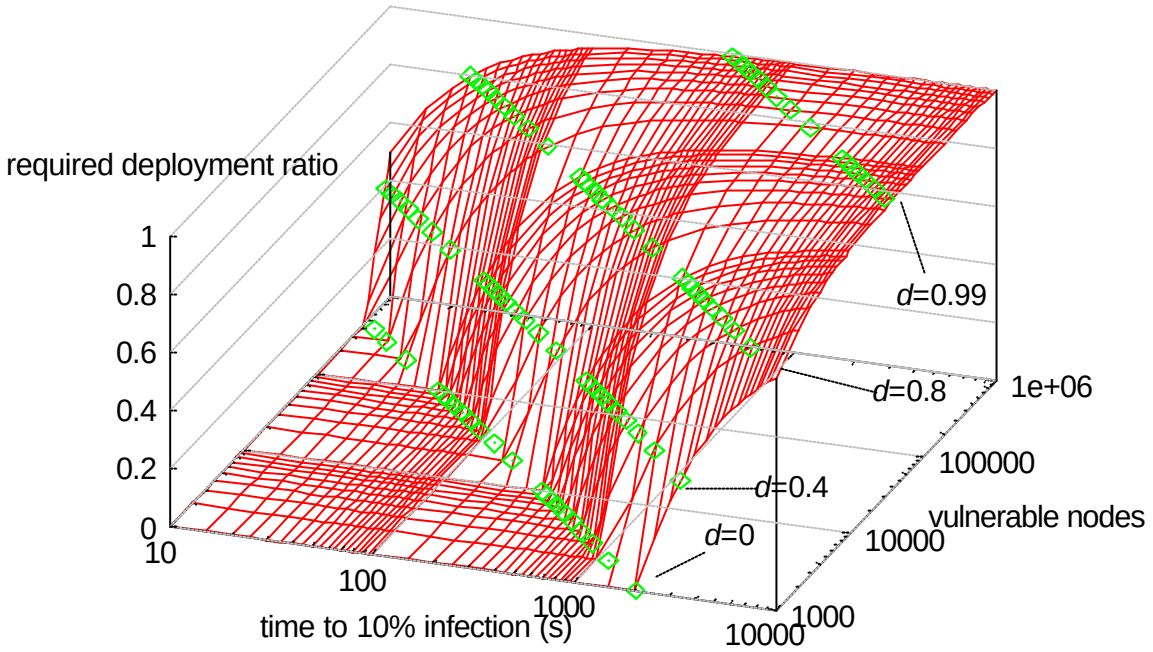

100000
10000
1000
100
10
1
time to 90% infection
100
1000
10000
100000
1e+06
number of vulnerable nodes

Figure 2


required deployment ratio
1
0.8
0.6
0.4
0.2
0
10
100
1000
time to 10% infection (s)
1000
10000
vulnerable nodes
100000
1e+06
d=0.99
d=0.8
d=0.4
d=0