Is Early Warning of an Imminent Worm Epidemic Possible?

Hyundo Park, Hyogon Kim, and Heejo Lee, Korea University

Abstract

This article introduces a novel anomaly detection method that makes use of only matrix operations and is highly sensitive to randomness in traffic. The sensitivity can be leveraged to detect attacks that exude randomness in traffic characteristics, such as denial-of-service attacks and worms. In particular, we show that the method can be used to alert of the imminent onset of a worm epidemic in a statistically sound manner, irrespective of the worm's scanning strategies.

Network intrusion detection is performed by monitoring network traffic and detecting the evidence of attacks by scanning known signatures (misuse detection) or recognizing anomalous traffic behaviors (anomaly detection). Misuse detection has been widely used for finding known attacks, and the low false alarm rate is one of its biggest advantages. While misuse detection is effective against known attacks, it cannot cope with freshly devised attacks whose signatures are not yet known. Anomaly detection, in contrast, has strength in detecting unknown attacks. But the cost is that the false positive probability is generally high. So typically, additional information is applied to reduce false alarms in anomaly detection, where the specific information to be used varies depending on the given circumstances.

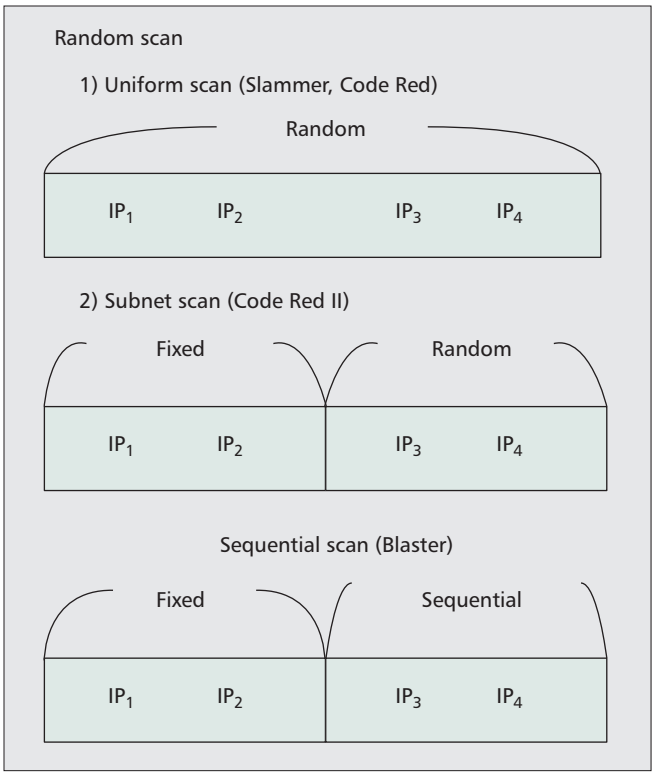
Existing anomaly detection mechanisms, as they are applied to the problem of worm detection, can be classified into three approaches: examining a sudden increase in new connection attempts, examining a sudden increase in connection failures, and examining a sudden increase in abnormal connection attempts. The first approach detects unknown worms by counting the number of new connection attempts [1]. For the traffic under watch, it tracks the network-level activity information in terms of distribution of source addresses, destination addresses, and ports, among others. If the information suddenly changes, this scheme raises an alarm. The second approach detects fresh worms by counting the number of connection failures, typically signaled by Transmission Control Protocol (TCP) reset packets, Internet Control Message Protocol (ICMP) destination unreachable messages, or TCP timeouts. As general worm scans target a large population of hosts to infect as many vulnerable hosts, scanning packets abound when worms are active. But not a few of the connection attempts produce connection failures [2]. So the approach detects worms by monitoring such an indication. The third approach detects novel worms by counting the number of abnormal connection attempts, such as those not using Domain Name System (DNS). This is because most legitimate users trigger Internet-based services by domain names instead of Internet Protocol (IP) addresses [3].

Recent advances in anomaly detection techniques that are general enough to be applied to different circumstances include the use of entropy, which is a measure of the irregularity of network traffic [4]. Entropy-based approaches can effectively detect excessive changes in network traffic. But when a large number of flows are aggregated at a monitored network, only attacks with big changes can be detected, whereas low-intensity attacks are not recognized properly. Another approach is attack visualization [5], which attempts to display a particular pattern of image under the occurrence of an attack in an intuitive way. Internet worms, distributed denial of service (DDoS) attacks, and network scans can be visualized as geometric shapes such as straight lines or rectangles on a 3D graph [5], or as triangles or diamonds on parallel coordinates [6]. Here also, only large-scale Internet attacks can be displayed when their activities contribute a noticeable amount of anomalous traffic.

In this article we attempt to devise a scheme that depends less on attack traffic amount and more on attack traffic quality so that we can pinpoint the attack with higher sensitivity, irrespective of relative attack traffic intensity. To this end we propose a novel metric for anomaly, the rank of the traffic matrix. A nice mathematical property of the matrix rank is that it is extremely sensitive to the randomness in the given matrix. Exploiting this fact and that the major modes of attacks such as DoS or worms or scanning necessarily involve randomness (or lack thereof) in the IP addresses, we devise the desired anomaly detection scheme. The scheme first populates a traffic matrix with ongoing network flows, and performs simple matrix operations in order to exclude most legitimate traffic. Then it computes the rank of the result matrix, which is so sensitive that with even a slight hint of randomness in the traffic it jumps close to the maximum.

Another pleasant property of the matrix rank sensitivity is that it is statistically sound, so we can build interesting applications on it. One demonstrated in this article is an early warning system for an imminent global worm epidemic. In the age of all-automated attacks, the precious few tens of seconds before the full inflammation of the epidemic is vital. For instance, in one realistic situation we show that the matrix rank already begins climbing when the worm-infected population is barely noticeable, and that at only 3 percent infection the rank hits the ceiling (well before the infection curve hits the exponential ride). Again, the rank as an early indicator of an imminent worm epidemic has a low false alarm rate thanks to its mathematical property, and precautionary countermea-

0890-8044/09/$25.00 © 2009 IEEE



■ Figure 1. *The randomness in IP addresses depending on scanning strategy.*

sures such as rate limiting can be introduced early on with less risk.

The matrix rank works both ways. In the presence of too much regularity in traffic, which is also a sign of attacks such as sequential scanning, the rank value nose dives. High or low, it signals abnormality, and renders the early warning system work irrespective of the scanning strategies of the worm.

Finally, the proposed approach does not require any a priori domain knowledge or training session to differentiate between normal and attack traffic patterns. Our experiences with normal real-life Internet traffic without worm activity show us that the rank hovers only in the middle range.

Matrix Rank as a Randomness Measure

Since a random binary matrix holds a high rank value [7], the matrix rank has been widely used for testing the quality of a random number generator such as the Diehard battery of tests [8]. Incidentally, many interesting attacks also increase the randomness in the source or destination IP address fields, e.g. scanning activities of Internet worms or packet flooding under a DDoS attack. The randomness either is intentionally injected to evade detection or obfuscate traceback, or naturally arises due to the distribution of attack launching locations (compromised hosts) in the IP address space. The idea is thus to apply the matrix rank measurement on the *traffic* matrix to check if the traffic harbors attacks.

The rank of a matrix is the count of non-zero rows after applying Gaussian elimination. In other words, the rank of the matrix is equal to the number of leading 1s in the matrix after the treatment. In the case of a random $m_1 \times m_2$ binary matrix, the probability of the matrix whose rank is r is as follows:

$$2^{r(m_1+m_2-r)-m_1m_2} \prod_{i=0}^{r-1} \frac{(1-2^{i-m_1})(1-2^{i-m_2})}{(1-2^{i-r})}, \quad (1)$$

where $1 \le r \le \min\{m_1, m_2\}$ [7]. From Eq. 1 we claim that the rank value of a matrix is a reliable measure of randomness. For instance, if its rank is greater than or equal to 252, a 256×256 matrix is a random matrix with over 99.999 percent probability. So with a sufficiently high rank value, the false positive probability is extremely low. This is a highly desirable property of the matrix rank when it is used for anomaly detection whose biggest weakness is the high false positive rate.

Traffic Matrix Construction

To make traffic amenable to randomness check, we translate traffic into the form of a matrix. Specifically, IP addresses need to be represented in the matrix as they take on the randomness in the attack traffic. Also, to catch Internet worm activity, we need to look at the destination IP address of the passing traffic since the worms select subsequent targets randomly [9]. Today's Internet worms use a random number generator to maximize their propagation speed and at the same time evade detection.[1]

In designing the traffic matrix construction scheme, we need to consider the fact that the four octets in an IP address could have separate dynamics depending on the particular strategy employed by the worm in action. Internet worms have used different scanning strategies. Let us denote each octet of an IP address as IP_1, IP_2, IP_3, and IP_4, respectively, such that

$$IP = IP_1.IP_2.IP_3.IP_4. \quad (2)$$

Slammer and CodeRed use the random scan strategy, which selects all four octets (IP_1, IP_2, IP_3, and IP_4) of the next target randomly. This strategy is also called a *uniform scan*.

CodeRed II, on the other hand, uses a scanning strategy with local preferences, which is called a *subnet scan*. The worm performs a random scan with probability 1/8. With probability 1/2, it will stay with the same IP_1. With probability 3/8, it will stay with the same $IP_1.IP_2$. Thus, the scanning strategy of CodeRed II is fully random in IP_3 and IP_4, but partially random in IP_1 and IP_2. Blaster selects one $IP_1.IP_2$ randomly, then sequentially scans subsequent targets within the Class B network until it selects another target network. So IP_1 and IP_2 are random, but the distribution of IP_3 and IP_4 is sequential. Notice, however, from the perspective of the attacked network the distribution of destination IP addresses of the scanned traffic would likely look sequential, not random. Figure 1 illustrates the scanning strategies of these worms.

Below, for convenience, we call the uniform scan and subnet scan simply *random* scan, as they have randomness in some parts of IP addresses. We use the classification of random vs. sequential for the scanning strategies of Internet worms.

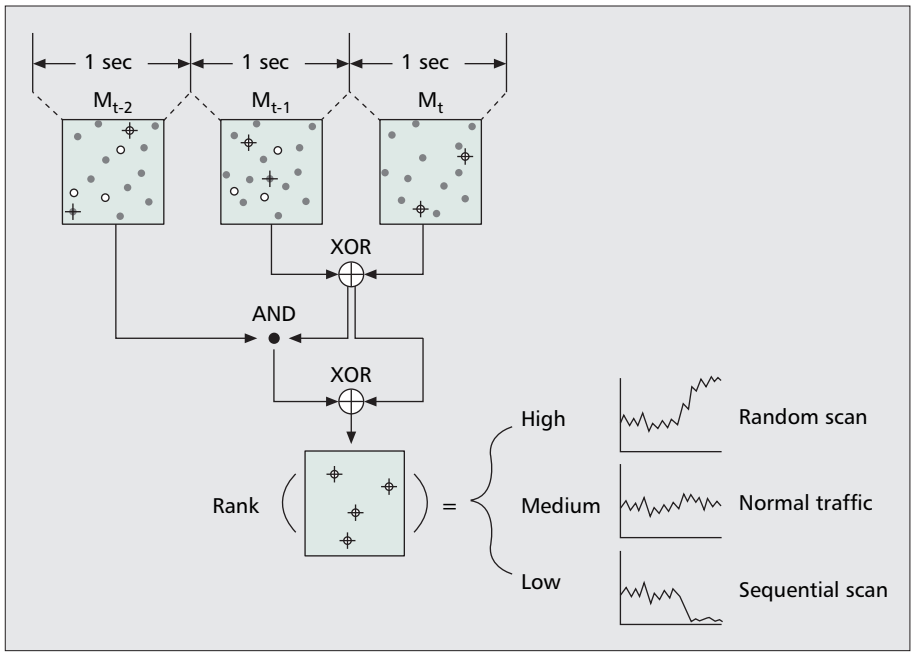
Let us consider a 256×256 binary matrix, which requires only a small memory space (8 kbytes) to store. The matrix can represent 65,536 distinct destination IPs at maximum. In order to preserve the randomness in the IP addresses used by a worm, we need to map the IP addresses to the corresponding bit in the matrix in a clever manner. Given i and j for the row and column index, respectively, we define the mapping as

$$i = IP_1 \oplus IP_3, j = IP_2 \oplus IP_4, \quad (3)$$

[1] *Although not addressed in this article, the same idea can be applied to DDoS attack detection. The difference is that source, not destination, addresses should be used in the matrix construction. The randomness in the source IP address arises due to random infection by malicious codes such as botnets, or the source IP address spoofing for traceback obfuscation [10].*

where ⊕ denotes the bitwise exclusive OR (XOR) operation. Notice that high randomness is preserved in the scanning strategies adopted by the aforementioned real-life worms. In the case of sequential scan worms, on the other hand, the mapping function performs a permutation due to the XOR, but once the mapping selects each row (XORed IP_3), it fills the row with 1s in a permuted sequence (XORed IP_4). In the end the row has all-1 entries, which contributes little to the rank. So the lack of randomness is also preserved.

Other conceivable scanning strategies, albeit not used by real-life worms yet, might require different mapping functions or even multiple matrices. But the focus of this article is on illuminating the potential of the matrix rank as a high-quality attack indicator, so we simply assume in the discussion below that the mapping has been done in a randomness-preserving way.



■ Figure 2. *The mechanism of filtering and rank measurement for anomaly detection.*

Once the mapping is defined, the construction of the traffic matrix is easy. At the beginning of each unit period, the traffic matrix is filled with zeros. Then for each destination IP address in the passing flow, the corresponding entry is overwritten with value 1. The matrix continues to be filled with 1s this way until the unit period ends. Here, the length of the unit period would depend on the working environment, but typical values are on the order of seconds, such as 1 s, 10 s, and so on.

Traffic Filtering Matrix Operations

As a traffic matrix construction is completed for a given unit duration, legitimate traffic should be as much removed from the matrix representation as possible for precise attack detection. One major advantage of our approach for the sake of filtering comes from the use of a matrix, which simplifies the operation. Specifically, the bitwise XOR operation on two traffic matrices from consecutive time units eliminates most legitimate flows, and mainly suspicious traffic remains at the result matrix. Also, we can pan out long-lived legitimate flows by the bitwise AND operation on two consecutive matrices.

Let M_t denote the matrix constructed during time unit t. Then $M_t \oplus M_{t-1}$ represents the matrix after the XOR operation of the corresponding entries in M_t and M_{t-1}. The XOR operation is to remove the overlapping entries, which are usually not malicious. For instance, when there are four legitimate flows, and one packet per unit time is generated by a worm, the XOR operation on the simplified 4×4 matrices will return the filtering result as follows:

$$\begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \oplus \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & \mathbf{0} & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Note that the rank of matrix $M_t \oplus M_{t-1}$, $R(M_t \oplus M_{t-1})$, is likely to be smaller than $R(M_t)$ unless there is a surge of new legitimate flows at time t. In contrast, we will have a higher rank after the XOR operation if random scan traffic increases, that is, $R(M_t \oplus M_{t-1}) > R(M_t)$.

Even after the XOR, some legitimate flows will leave their traces in the matrix, such as those that newly start at t and those that terminate at $t - 1$. In order to exclude even these, we perform the following matrix operations:

$$M_t' = M_{\text{XOR}}(t) \oplus (M_{\text{XOR}}(t) \cdot M_{t-2}), \quad (4)$$

where $M_{\text{XOR}}(t)$ represent the XOR operation of two consecutive matrices M_t and M_{t-1}, that is,

$$M_{\text{XOR}}(t) = M_t \oplus M_{t-1}.$$

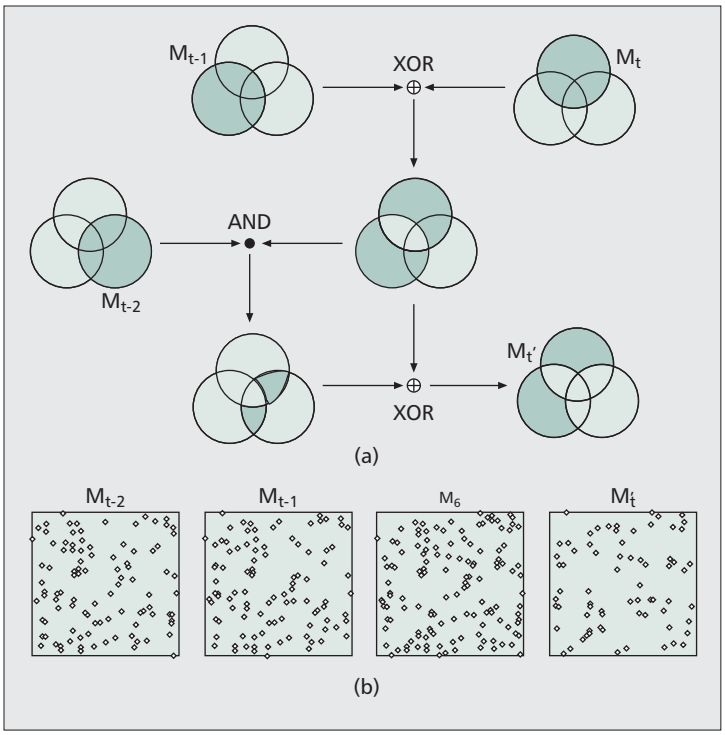
Then $M_{\text{XOR}}(t) \cdot M_{t-2}$ retains the terminating legitimate traffic at t and $t - 1$, which M_t' purges. Figure 2 pictorially depicts the filtering mechanism.

Starting connections at time t may or may not be legitimate, which we cannot decide until $t + 1$ so that they are not excluded from M_t'. The Venn diagram representation of M_t' is given in Fig. 3a. It shows that the flows that span $t - 2$ and $t - 1$ as well as those that span $t - 1$ and t are being eliminated as they are likely legitimate.

Figure 3b visually illustrates the effect of the operation in Eq. 4, on a /16 campus network traffic trace where the average number of clients and packets are 132 and 1847 during the time span of 187 s (of which only three consecutive seconds worth of traffic was used to generate the figure). In the figure we obtain $R(M_t) = 99$ and $R(M_t') = 53$. As the figure demonstrates, the use of the filtering operation causes the rank to decrease significantly across the entire trace, where the refined rank value carries more meaningful information as to the traffic randomness quality. One of the charms of this simple matrix filter is that it is stateless. We do not need protocol-specific information such as TCP/IP headers or packet contents for anomaly detection. We believe that the statelessness of the proposed scheme contributes to its usability.

Ranks from Normal Real-Life Traffic

In order to demonstrate that the rank value of normal traffic indeed remains under that of the random matrix, we comput-



■ Figure 3. *The effect of matrix operations with XOR and AND: a) the region of M_t' with respect to Mt, Mt-1, and Mt-2; b) the effect of matrix operations on the traffic matrix in a /16 network.*

ed $R(M_t')$ for some publicly available real-life traces (Fig. 4). We first ensured that there had been no worm epidemic or strong worm activity reported during the given trace collection time. Also, we included traces that contain a flash event, an extremely concentrated traffic spurt, to check if the randomness-based anomaly detection works under such instability.

The first trace is from two trans-Pacific T-3 links connecting the United States and a Korean Internet gateway between 9:36 a.m. and 9:55 a.m. on December 14, 2001. From this trace we extract two 90-s subtraces that contain a flash event situation, called JScript and WinUpdate, respectively. The first one shows numerous clients sending a huge number of requests to a server to download newly issued versions of Java scripts for their personal Websites and blogs.[2] The second one contains requests to a Microsoft Windows update Website that attracted a huge number of requests when an accumulated patch to Windows Internet Explorer was released.

The second trace is from the CAIDA archive,[3] and records the traffic from an Internet service provider (ISP) located at the Equinix data center in Chicago, Illinois connected to Seattle, Washington through an OC-192 pipe. We extract two 20-s subtraces in this traffic, denoted CAIDA01 and CAIDA02. The average numbers of new connections and packets per second of CAIDA01 and DAIDA02 are 19, 2, and 988, 729, respectively. And the total number of packets during 20 s is 19,754 and 14,588, respectively. The number of new connections has been manually obtained by counting the TCP SYN packets. So the number of new connections is much smaller than the number of total packets. We notice that the rank value generally rises with traffic intensity, but even the OC-192 traffic does not push it near the rank of a random matrix.

Sensing Imminent Worm Epidemics

Now, let us examine the dynamics of the traffic matrix rank in the face of worm activity. For simplicity of illustration, let us assume random scanning worms. Let N denote the vulnerable population size for this particular worm, and L the monitoring network size (in number of IP addresses). Furthermore, let α denote the infection ratio, so αN is the number of infected hosts in the vulnerable population. Finally, let the scan rate of worm β be the number of scans per second per worm.

Suppose we see some global worm epidemic on a scale comparable to Slammer, which we simulate here. In the simulation we set $N = 10^6$, $\beta = 10^3$, and $L = 2^{16}$. Note that the value of β is not excessive considering that the scan rate of the Slammer worm was observed to be 26,000 scans/s at maximum, and approximately 4000 scans/s/worm on average [11]. Figure 5 gives the simulation results, and shows how sensitive the rank is compared to the number of infected hosts. When only 3, 5, and 10 percent of vulnerable hosts are infected at 1000, 600, and 300 times the scan rate of the Internet worm, respectively, the rank value already approaches the maximum, exceeding 252. Thus, we get a rank over 252 when α =0.03 when $N = 10^6$, $\beta = 10^3$, and $L = 2^{16}$.

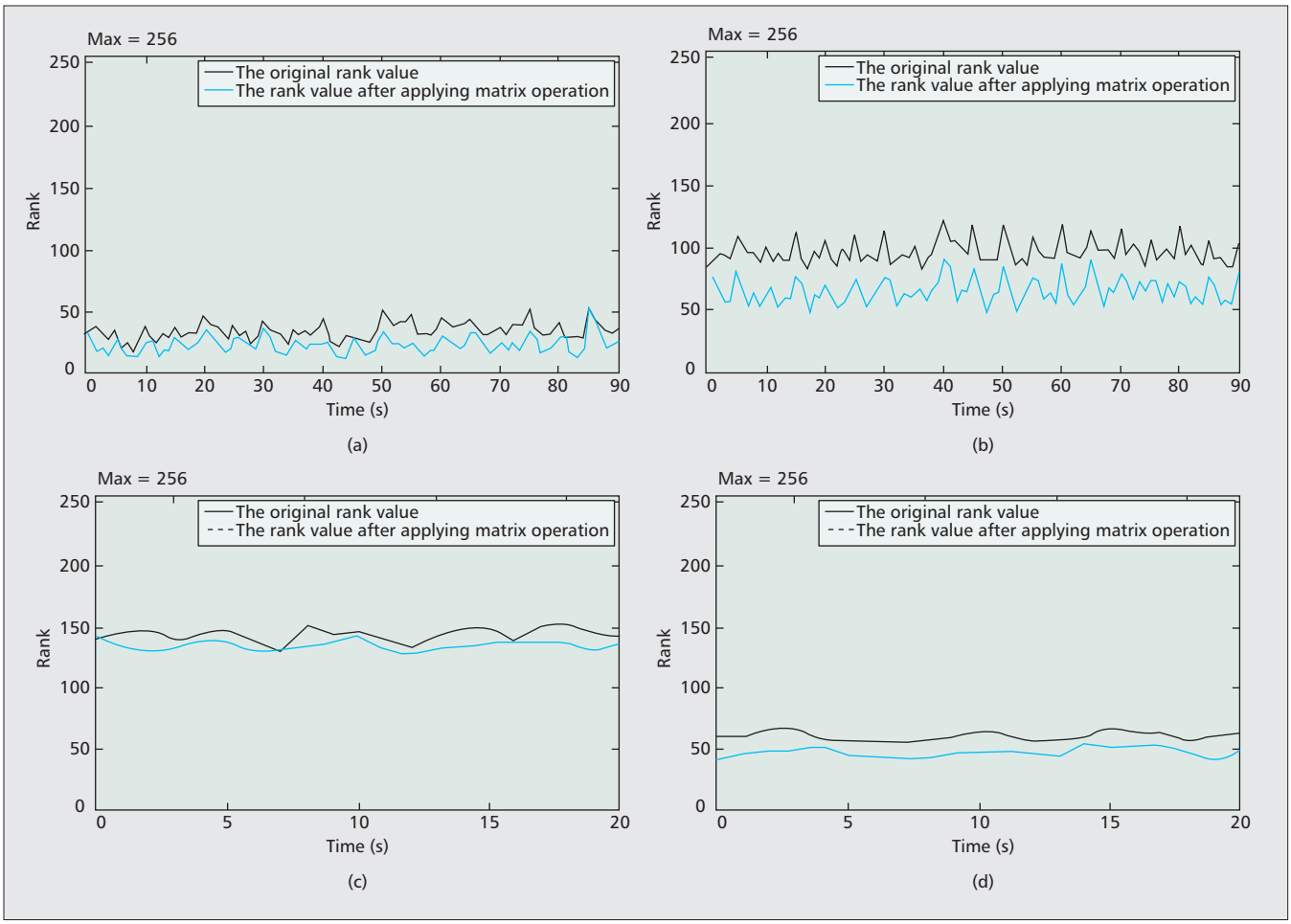
In the given setting we can definitely detect an epidemic if a total of more than 32 million scanning packets crisscross the Internet per observation interval. Then approximately 488 scans hit a /16 network, and the rank value will always exceed the randomness threshold. At this point, the ratio of the infected hosts would be 32/β. So if the scan rate β of a single worm is over 32 but not large, the detection time can get large in terms of the infected population. But then the time interval to monitor the network in our approach can be flexibly changed, such as to 10 s, to cope with the problem. For example, if the Internet worm scans only 10 hosts/s, if we elongate the monitoring interval to 10 s, the proposed method can detect the Internet worm when 32 percent of total vulnerable hosts are infected by the Internet worm.

In the above particular example, the instance of the rank rising to an otherwise impossible value is just before the epidemic explodes into its exponential growth phase. In today's world of all automated attacks, human intervened countermeasures are getting too slow to stop an epidemic. For instance, Slammer effectively grew to a full epidemic within 10 minutes. The buffer time of a few tens of seconds, hard won by the early warning indicator such as the traffic matrix rank we propose in this article, could be precious to automatically deploy precautionary countermeasures such as rate limiting. Such damping efforts could prevent an epidemic or at least procrastinate its development so that a more adequate prescription can be injected.

Figure 6a shows the rank values as a function of scan rate β for the cases of two different scanning worms. The rank is measured when 10,000 hosts were infected (α = 0.01), where worm epidemics are injected into the real network trace captured in the gateway of a /16 university campus network. In the case of a random scan worm, the rank increases dramatically as the scan rate increases. In the case of a sequential scan worm, the rank suddenly drops after β =1000. This property enables us to detect non-uniform scan worms such as Blaster by monitoring a single block of address space, whereas previous worm monitoring approaches are effective only when their monitoring address space is largely distributed [12]. In essence, as the worm activity intensifies, it drives the matrix

[2] *http://files.cometsystems.com*

[3] *http://caida.org*



■ Figure 4. *The various rank values under FE: a) Jscript; b) WinUpdate; c) CAIDA01; d) CAIDA02.*

m	32	64	128	256	512	1024
γ	0.063	0.041	0.025	0.014	0.008	0.005

■ Table 1. *The random factor γ for the size of traffic matrix.*

rank either extremely high or extremely low, a clear indication. Otherwise, the rank hovers at non-extreme values, as shown in the figure.

In Fig. 6a, randomly scanning packets drives the matrix rank extremely high, whereas sequentially scanning packets drives it extremely low. The reason for the latter is the work of Gaussian elimination in computing the matrix rank. If all elements of some rows become 1 (e.g., by a sequentially scanning worm), such rows except one become all 0 after Gaussian elimination, as shown below in a simplified example.

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Therefore, if a sequential scanning worm propagates on the Internet, the matrix rank dips extremely low.

The rank shrinking effect by the sequential scanning worms is so strong that if both sequentially and randomly scanning worms simultaneously propagate, the matrix rank is completely dominated by the sequentially scanning worm because we overwrite the connection information of traffic on the matrix. The rank never stays in the middle, which would mean no detection of either type of worm. Figure 6b shows that no matter how many randomly scanning worms come, the matrix rank dips due to a concurrently active sequential scanning worm.

Figure 6a shows that the two different types of worms experience the rank value change at a similar range of scan rates. The detection time for the worms would be similar and be a function of the scan rate, but not the scan type.

In order to alert of the imminent onset of a worm epidemic, how many scanning packets would we need to collect in the matrix to grow the rank over a large threshold, say 252? We can calculate the answer as follows. Let γ denote the random factor so that the random sprinkling of γm^2 number of 1s would form a random binary matrix with an extremely high rank value. Then for a random scanning worm to register enough 1s in the traffic matrix to turn it random, one can argue that we need γm^2 scanning packets. But in fact, the required number of random scanning packets is less. Notice in Fig. 2 that scanning packets have the effect of approximately doubling the non-zero entries in M_t' after $M_t \oplus M_{t-1}$ operation. Thus, we only need half of $\gamma \cdot m^2$ to build a random $m \times m$ matrix. Thus, we obtain the following relation among the parameters as the answer to the above question:

$$\alpha N \cdot \beta \cdot L/2^{32} \geq \gamma \cdot m^2/2. \quad (5)$$

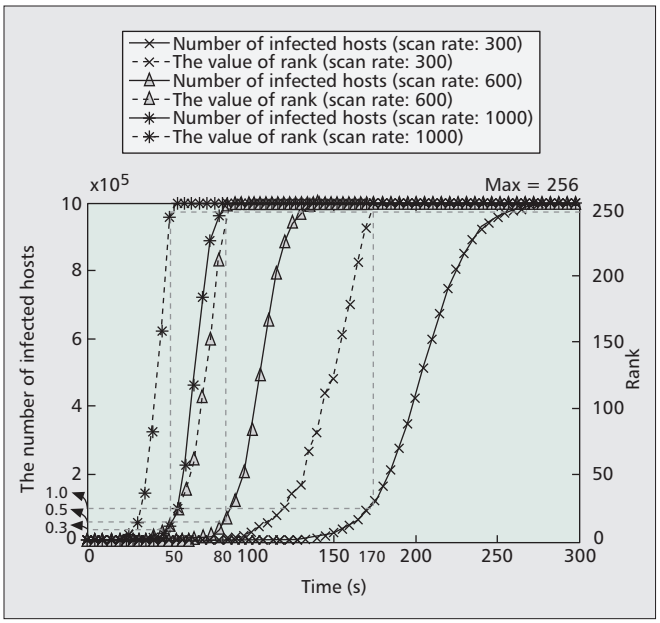
In other words, the values of N, β, α, and L will determine whether the resulting matrix M_i' is random or not. Here, the parameters N and L are given by the host and network configurations, β is largely the worm's attribute, and m is given by the anomaly detection module configuration. The remaining parameter γ, on the other hand, is a property of matrix randomness and only depends on m. We can experimentally measure the average γ through iterations of random matrix construction, which is shown in Table 1. We can see that only a small fraction of the traffic matrix estate needs to be turned on by random scan packets to enable randomness detection through the rank. It shows the sensitivity of the rank metric from another angle, as it is used in detecting randomness.

The inequality of Eq. 5 lets us understand the interplay between the parameters better. In particular, we can see that earlier detection is possible if α can be lowered, say, by increased β. For instance, in Fig. 6a $\alpha = 0.03$ for $\beta = 1000$, whereas in the same condition α decreases to 0.01 with $\beta = 3000$. This inverse proportionality between α and β (also evident in Eq. 5) lets us detect faster worms faster, which is a desirable property of the proposed detection scheme.

Finally, according to Eq. 5, the infection ratio at which the epidemic is first detected by matrix rank is only about 0.3 percent $\sim$ 1.2 percent when $N = 10^5 \sim 10^6$ and $\beta = 5000 \sim 10{,}000$. In essence, for the parameter ranges that could lead to a global worm epidemic, the proposed method can give us an early warning to the rising danger when there is still time to react.

Conclusion

A matrix is a convenient data structure with many well defined powerful operations. In this article, we introduce a matrix-based intrusion detection mechanism, which constructs a matrix from network traffic and measure the rank of the matrix after matrix operations for legitimate traffic filtering. From the experiments and analysis, we have shown that the matrix rank is a highly effective and reliable measure to sense the imminent onset of an Internet worm, regardless of the worm's scanning strategy. This matrix approach can be useful to detect not only Internet worms, but also other types of attack that increase the randomness in network traffic.



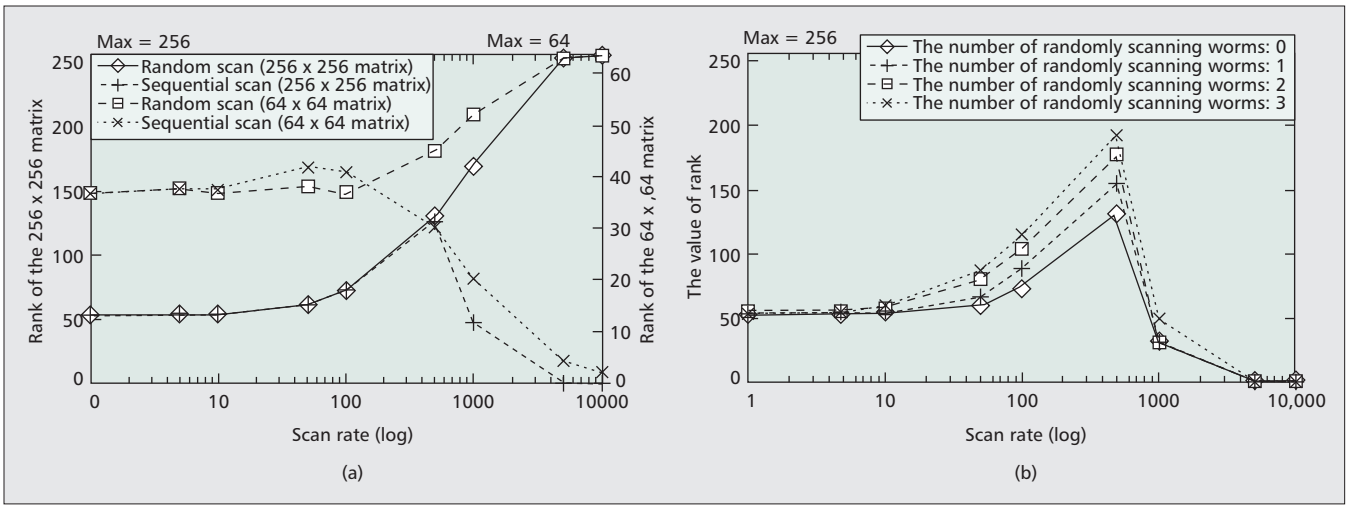
■ Figure 5. *The number of infected hosts and the corresponding rank as a function of time.*

Acknowledgment

This work was supported in part by the ITRC program of the Korea Ministry of Knowledge Economy (MKE), the IT R&D program of MKE/IITA(2009-S-026-01), the Defense Acquisition Program Administration and Agency for Defense Development, and the Korea Research Foundation Grant 2009-0080413.

References

[1] C. Leckie and R. Kotagiri, "A Probabilistic Approach to Detecting Network Scans," *8th IEEE/IFIP Net. and Ops. Mgmt. Symp.*, Apr. 2002, pp. 359–72.
[2] J. Jung *et al.*, "Fast Portscan Detection Using Sequential Hypothesis Testing," *IEEE Symp. Sec. and Privacy*, 2004, pp. 211–25.
[3] D. Whyte, E. Kranakis, and P. Oorschot, "DNS-based Detection of Scanning Worms in an Enterprise Network," *12th Net. and Distrib. Sys. Sec. Symp.*, 2004.
[4] W. Lee and D. Xiang, "Information-Theoretic Measures for Anomaly Detection," *IEEE Symp. Sec. and Privacy*, 2001.
[5] H. Kim, I. Kang, and S. Bahk, "Real-Time Visualization of Network Attacks on High-Speed Links," *IEEE Network*, vol. 18, no. 5, Sept. 2004, pp. 30–39.
[6] H. Choi and H. Lee, "PCAV: Internet Attack Visualization on Parallel Coordinates," *ICICS 2005*, LNCS, vol. 3783, Dec. 2005, pp. 454–66.



■ Figure 6. *The rank values of sequentially or randomly scanning schemes: a) rank values as the function of a scan rate; b) rank values as the function of a scan rate.*

[7] G. Marsaglia and L. H. Tsay, "Matrices and the Structure of Random Number Sequences," *Linear Algebra and Its Apps.*, vol. 67, 1985, pp. 147–56.
[8] G. Marsaglia, "Diehard: A Battery of Tests of Randomness," 1996; http://www.stat.fsu.edu/pub/diehard/
[9] H. Park, H. Lee, and H. Kim, "Detecting Unknown Worms Using Randomness Check," *IEICE Trans. Commun.*, vol. E90-B, no. 4, Apr. 2007, pp. 894–903.
[10] H. Park *et al.*, "Distinguishing between FE and DDoS Using Randomness Check," *Info. Sec. Conf., LNCS*, vol. 5222, Sept. 16. 2008, pp. 131–45.
[11] D. Moore *et al.*, "Inside the Slammer Worm," *IEEE Security & Privacy*, vol. 1, no. 4, July/Aug. 2003, pp. 33–39.
[12] C. C. Zou *et al.*, "The Monitoring and Early Detection of Internet Worms," *IEEE/ACM Trans. Net.*, vol. 13, no. 5, Oct. 2005, pp. 961–74.

Biographies

HYUNDO PARK (hyundo95@korea.ac.kr) received B.S., M.S. degrees from the Department of Computer Science and Engineering, and also received a B.S. degree from the Department of Mathematics at Korea University, Seoul. He is currently working toward a Ph.D. degree in the Department of Computer Science and Engineering at Korea University.

HEEJO LEE (heejo@korea.ac.kr) is an associate professor at Korea University. Before joining Korea University, he was at AhnLab, Inc. as a CTO from 2001 to 2003. He received his B.S., M.S., and Ph.D. degree from POSTECH, Pohang, Korea, and was a postdoctorate researcher at Purdue University. He serves as an editor of the *Journal of Communications and Networks*. He has been an advisory member of the Korea Information Security Agency and Korea Supreme Prosecutor's Office. With the support of the Korean government, he worked on constructing the National CERT in the Philippines (2006) and consulted on cyber security in Uzbekistan (2007) and Vietnam (2009). More information is available at http://ccs.korea.ac.kr.

HYOGON KIM (hyogon@korea.ac.kr) is a professor at Korea University. He got his Ph.D. from the University of Pennsylvania in 1995. Prior to joining Korea University, he was a research scientist at Bell Communications Research (Bellcore). His research interests include Internet protocols and applications, wireless networking, network security, and computational neuroscience.