An Efficient Signature-Based Approach for Automatic Detection of Internet Worms over Large-Scale Networks

Kumar Simkhada[1,*], Tarik Taleb[1,*], Yuji Waizumi[1,*], Abbas Jamalipour[2], Nei Kato[1,†], and Yoshiaki Nemoto[1,*]
[1] Graduate School of Information Sciences
Tohoku University, Japan
*{kumar, taleb, wai, nemoto}@nemoto.ecei.tohoku.ac.jp
†{kato}@it.ecei.tohoku.ac.jp
[2] School of Electrical & Information Engineering
University of Sydney, Australia
a.jamalipour@ieee.org

***Abstract*— Internet Worms pose a serious threat to today's Internet. Signature matching is an important approach to detect worms. However, as most signature development processes are manual, they require significant time. They are thus not efficient in reducing the damage worms may cause. In this paper, an efficient signature-based method is proposed for automatic detection of worms over large-scale networks. In the proposed system, detection is performed in a hierarchical manner. Security managers of local networks collect worm-like or suspicious flows and handle these flows to high-hierarchy metropolitan managers. In response, the latter use this information to generate robust signature. The global manager which lies on top of the hierarchy, multicasts the signature to local managers via metropolitan managers. This enables local managers to detect worms that try to penetrate into their networks. The proposed system is evaluated using an off-line real network traffic that contains traces of worms. Experimental results indicate that the proposed system exhibits high detection rates with low false alarm rates.**

I. Introduction

Internet worms are one of the most devastating and distinct types of attacks in the Internet. When they infect a host, they can remove its system files or modify the contents of its applications. They can also implement Distributed Denial of Service (DDoS) tools and bundle in DDoS attacks. After infecting a host, they propagate to other *vulnerable hosts* using several techniques such as scanning, target lists, and passive monitoring [1]. Table I lists some recent worms and the number of hosts they infected [2]–[5]. The table indicates the vulnerability of present networks to worms and the limitations of current security systems. Given the increasing number of worm types, their fast propagation speeds, and the threat they present to the Internet resources, adequate measures to combat against worms is mandatory. Without such measures, cyber terrorism may deprive large enterprises from making efficient use of the Internet and may cause significant damages in terms of time, infrastructure, and human resources. Protection of the Internet against worms forms the focus of the research outlined in this paper.

Most current worm detection techniques rely on signature matching. They enjoy high popularity among network security systems as they exhibit higher detection rates and generate fewer false alarms compared to other methods. However, most worm signatures are manually developed and are thus costly due to the cumbersome work and time they require.

TABLE I
A list of some recent worms and their caused damage.

Worm	Year	Type	Number of infections
Code Red	2001	Scan	359 thousand in 14 hrs.
SQL Slammer	2003	Scan	75 thousand in 10 mins.
Blaster	2003	Scan	330 thousand in 5 days
Witty	2004	Scan	20 thousand in 1 hour
Beagle	2004	Email	504 thousand
NetSky	2004	Email	6.1 million
MyDoom	2004	Email	2.3 million

In addition, manual development of worm signatures is not efficient in reducing the damage worms may cause. Indeed, by the time experts develop a worm signature, significant damage may have already been caused. Furthermore, in case of polymorphic worms, new signatures have to be developed for each variant. In order to detect worms at early phases of their propagation, generation of worm signatures has to be performed in an automatic manner.

In this paper, we present a system to automatically generate highly accurate worm signatures over large-scale networks. In the proposed system, worm detection is performed in a hierarchical manner. The network topology is divided into a number of metropolitan areas. Each metropolitan area comprises a number of local networks. A global security manager controls the whole network topology and directly communicates with metropolitan security managers located at each metropolitan area. When a *worm-like* traffic is detected at a local network by its local security manager, the latter qualifies the traffic as *suspicious* and transmits it to the metropolitan manager of its corresponding metropolitan area. The metropolitan manager analyzes then the received suspicious traffic and decides whether a worm is spreading in its monitored area. Accordingly, it generates a highly accurate signature for the detected worm, and passes it to the global security manager. The global manager then relays the signature to local security managers via metropolitan managers. This operation helps to hinder the further spread of the worm and its variants.

The remainder of this paper is organized as follows. Section II showcases the propagation methods used by worms and surveys the state-of-the-art in the area of worm detection. Section III provides a detailed description of the proposed scheme. The performance of the proposed scheme is evaluated

1-4244-0355-3/06/$20.00 (c) 2006 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

in Section IV. Finally, Section V concludes the paper and discusses possible future work.

II. Related Work

To detect worms, a number of systems have been proposed in recent literature. Systems such as HoneyStat [6] and Honeycomb [7] use honeypots to detect worms. Honeypots have the ability to test the presence of worms in real traffic. However, they can detect worms only when worm packets are sent to them. In case of highly interactive honeypots, which leave their OS entirely exposed to intrusions, worms may significantly compromise the honeypots and use them to attack other targets. Therefore, there is a significant risk in using honeypots. Data mining is a frequently used method for detecting email worms [8]. This method is effective for the detection of unknown intrusions including worms. However, it can be costly in terms of time and system resources. [9]–[11] use taint analysis to detect buffer overflow attacks which are carried out by many notable worms such as Code Red. The basic concept behind taint analysis-based schemes is to retrieve inputs from unreliable sources and to track down the records of data affected by such inputs.

A system which matches destination port numbers between incoming and outgoing connections is proposed in [12]. This method is based on the fact that, when a worm infects a machine via a certain port, the machine sends worm packets more likely via the same port to other hosts. Whilst this may be efficient in detecting fast-spreading worms, it requires a massive amount of information storage. MET, Malicious Email Tracking, filters worm attachments from emails [13]. It assumes that worm attachments do not change during propagation and computes MD5 hash for every binary attachments. This assumption does not hold in case of polymorphic worms such as Mimail whose attachment files can change in each propagation step. [14] presents an architecture design of a "feedback email worm defense system" to protect email users in enterprise networks. The system consists of an "Email Service Unit" which directly sends "safe" emails to the users. It saves suspicious attachments at another server where the likeliness of a worm attack is statistically evaluated. If an attachment is not cleared, it is sent to a honeypot where the attachment's contamination with a worm is tested. This system has a credit of achieving low false alarm rates. It, however, requires long period of time to confirm the possibility of a worm attack. In [15], *Akritidis et al.* propose a content-based detection scheme to detect worms. Their scheme bases its detection on invariant portions present in the payloads of a worm. While their system targets only worm packets originating from clients to servers irrespective of their destination networks, our proposed scheme is designed to protect local networks from worms coming from outsiders. Additionally, unlike their system which uses Rabin fingerprints to identify the invariant portions, the proposed scheme uses tokens of fixed length.

Autograph [16] and Earlybird [17] are also two recent content based novel worm detectors. Both of these systems look for repetitive contents in worm payloads to generate signatures by using Rabin fingerprints. These systems generate single substring signatures. They assume that there exists a single substring that occurs in every worm payload but not in any normal ones. Polymorphic worms, which may alter parts of their payloads in their variants, can easily evade detection by replacing the *signatured* portions. Polygraph [18] uses multiple portions that are present in worm payloads. It generates conjunction signature, token-subsequence signature, and Bayes signature. In contrast, the proposed scheme extractss common tokens present in worm flows and differences them with normal flows to generate signature. It then reduces minimizes the rate of false negatives by checking the percentage of substrings present in test flows.

The above-mentioned techniques are deployable at only local networks. Most of these security systems do not address the need of information exchange to defend against globally propagating worms. Thus, for a robust and efficient defense against worms, a collaborative strategy among security systems of local networks is necessary. [19] proposes the collection of ICMP host unreachable packets from routers at the edge of an Internet Service Provider (ISP) to detect scans and control worms by blocking scanning hosts. However, due to privacy concerns, most routers are designed not to return ICMP host unreachable messages. Thus, this method can not be considered practical for global detection of worms.

DOMINO [20] and Indra [21] are two architectures which have been proposed to detect intrusions including worms in a distributed environment. In DOMINO, heterogeneous nodes share information on worms and make themselves aware of the presence of any worm in the network. DOMINO comprises also *active sink nodes* which verify the legitimacy of connections directed to unused IP addresses. On the other hand, in Indra, only interested and trusted peers in a Peer-to-Peer (P2P) network share information on any intrusion attempt directed at them. Compared to these two architectures, the proposed system adopts a hierarchical concept for information exchange.

III. A Large-Scale Worm Detection System

A. Components of the Proposed Architecture

Fig. 1 depicts the key components of the hierarchical architecture of the proposed system. It consists of several Metropolitan Area Networks (MANs) composed of a set of clusters of local networks. Each MAN comprises a metropolitan security manager. Metropolitan security managers communicate directly with Local Security Managers located within their service areas. Local managers search for flows carrying similar contents. They qualify them as *suspicious flows* and send them to their corresponding metropolitan managers. Metropolitan managers use cluster analysis to identify worms from the suspicious flows. They then generate worm signature by extracting common portions from the payloads. Metropolitan managers update worm events to the global manager which is situated on top of the hierarchy. The global manager uses these update information to determine the areas that are affected by worms. It then acknowledges the areas that are

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

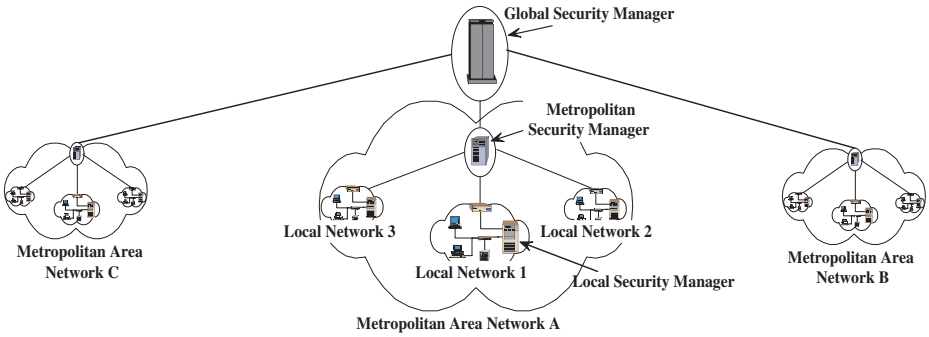


Fig. 1. The envisioned two-layer hierarchical worm detection architecture.

most likely to be affected by sending alert messages, along with the worm signature, to their metropolitan managers. By increasing the alert levels at targeted networks, worms can be detected at early stage, before they cripple the networks.

B. Collection of Suspicious Flows at Local Networks

The operation of local managers is depicted in the flowchart of Fig. 2. Local managers consist of two primary units, namely Signature Update Unit (SUU) and Anomaly Detection Unit (ADU). The SUU unit protects the local network from worms by using already available signatures. The ADU, on the other hand, collects worm-like flows from inbound traffic and sends them to the metropolitan manager. The working schemes of these units are described below.

1) Signature Update Unit (SUU): This unit functions in a similar way to existing Intrusion Detection Systems (IDSs) and firewalls. It filters the total network traffic and verifies its legitimacy by comparing it to the characteristics of some *normal traffic* and *attacking traffic*, already available at the IDS systems. Detected worms are blocked instantly while *normal traffic* is not hindered. It forwards the remaining traffic, or the *SUU filtrate*, to the ADU unit. The SUU unit is regularly updated with signatures relayed by the high-hierarchy metropolitan manager.

2) Anomaly Detection Unit (ADU): This unit analyzes the SUU filtrate and collects worm-like or suspicious flows. It then sends these suspicious flows to the metropolitan service manager. For this purpose, it exploits some intrinsic characteristics of worms. For instance, given the fact that a specific type of worm targets specific ports, analysis is carried out on a port basis. When a worm is actively propagating in the Internet, same character sequences appear in many flows. Unix commands and parts of executable programs may be examples of such information passed on to target hosts during each infection step. From each flow present in the suspicious traffic, the ADU unit extracts a fixed *number of sample tokens* (N_S) of constant length (L_S). It judges flows which contain *sample tokens* that occur in several flows as worm flows. In [22], the authors propose a scheme which differences worm flows and normal flows to extract sequences unique to worms. It is based on the fact that normal flows do not carry worm strings. This approach is taken in order to reduce the probability of normal tokens being selected as sample tokens.

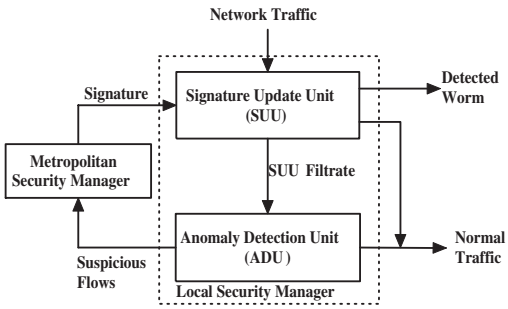


Fig. 2. Basic operations of the worm detection procedure at local security managers.

Let $\mathbf{T}$ be the set of these *sample tokens*. If a sample token $t_j \in \mathbf{T}$ exists in an inbound flow F_i, its *occurrence frequency* (f_j) is increased by 1. If only n_s ($0 \leq n_s < N_S$) sample tokens appear in an inbound flow, $(N_S - n_s)$ tokens are further extracted from such a flow and are added to $\mathbf{T}$. As tokens with high occurrence frequencies are likely to be parts of worm payloads, alerts are generated for flows that contain tokens whose occurrence frequencies exceed a predefined *Repetitive Occurrence Threshold* (Δ_{TH}). A flow for which an alert is generated is considered to be a *suspicious flow* and is sent to the metropolitan manager. To mitigate the memory overhead that may be incurred due to the saving of sample tokens, *old* sample tokens are gradually deleted from $\mathbf{T}$. Hence, only the sample tokens observed during a predefined *Token Caching Time* (θ_T) remain throughout the analysis.

C. Signature Generation at Metropolitan Managers

If a worm is spreading across several local networks, a metropolitan manager is likely to receive similar suspicious flows from its monitored local managers. The metropolitan manager sorts worms from these suspicious flows. It then uses the worm flows to generate a highly accurate signature.

Character distributions of payloads that belong to a particular worm are usually similar whereas those of normal payloads vary with each other by a great extent. This property is used to separate flows that belong to a certain type of worm from the rest of the suspicious flows. Let n be the number of suspicious flows collected at a metropolitan manager. For each suspicious flow, occurrence frequencies of the 256 ASCII codes are extracted. The values of the occurrence frequencies are further normalized and are used as coordinates of the suspicious flow in a 256 dimensions space. Each of these payloads is initially considered as a cluster. Nearest clusters are joined to form a new cluster until the total number of clusters becomes less than $(n/2)$. Clustering half of the received flows can be considered sufficient because a number of worm flows will be

already grouped inside a cluster by then. The biggest cluster is regarded as the worm cluster.

The flows that form the worm cluster, referred to as *worm flows* throughout this paper, are used for the signature generation. The latter is carried out in two phases is proposed in [22]. In the first phase, common substrings with a length longer than a pre-defined threshold (L_{MIN}) are extracted from worm flows. Among the extracted substrings, the ones appearing in normal flows are excluded. Hence, multiple substrings, each with a length longer than L_{MIN}, are generated. To form a signature out of these substrings, several approaches can be considered. Weighting the substrings, and combining some or all of them are two possible candidates. However, in a real network scenario, decisions regarding the use of signature substrings should be made instantly. We overcome this problem by using a new parameter, *Attack Tolerence Level* (ATL). ATL is defined as the minimum percentage of signature substrings that should be detected in a flow in order to generate an alarm for the flow. Use of multiple substrings in the signature, along with appropriate adjustment of ATL, make the proposed approach robust in detecting polymorphic worms.

IV. Performance Evaluation

A. Experimental Set-up

The following two quantities are used to investigate the efficiency of the proposed scheme:

1) **True positives** (N_{TP}): number of worms that are successfully detected.
2) **False positives** (N_{FP}): number of innocent flows for which alerts are mistakenly generated.

We describe separately the experimental set-ups for collecting suspicious flows and for investigating the performance of the generated signatures.

1) Collection of Suspicious Flows: An off-line real network traffic containing traces of Beagle worm is used to test the performance of a local manager. It consists of a total of 3054 inbound flows on port 25 directed to the monitored network which consists of 85 computer hosts and 160 actively used email addresses. Among the inbound flows, 34 flows contain worm information.

The required length of tokens-caching time (θ_T) depends on the type of the targeted worm. For fast spreading worms such as Slammer, setting θ_T to a low value (in the order of seconds) is sufficient to collect enough worm flows. However, in our case where we aim to collect inbound email worms, a reasonable length of time is required to gather worms. Hence, θ_T is fixed to 60 minutes.

2) Performance Evaluation of Signatures: A separate off-line real network traffic, captured after the traffic used for evaluation of local managers were captured, is used to investigate the efficiency of the signatures generated by metropolitan manager. This traffic consists of 45,193 flows destined to port 25, among which 271 are Beagle worms.

B. Collection of Suspicious Flows at a Local Network

In evaluating the performance of local managers, different scenarios are considered by changing the length of sample tokens (L_S), the number of sample tokens extracted per flow (N_S), and the Repetitive Occurrence Threshold (Δ_{TH}).

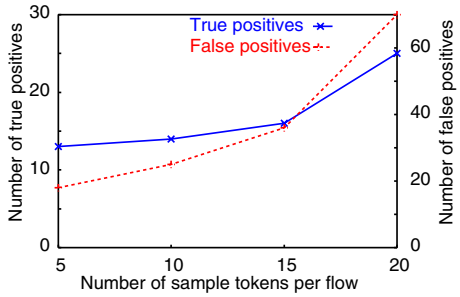
The payload distribution of a flow is better represented if a large number of sample tokens are extracted from its payload. Hence, a high detection rate is possible when N_S is set to a high value. However, this also increases the probability of extracting normal sequences as sample tokens, thus increasing the risk of false positives. This is depicted in Fig. 3(a) which shows a high number of true positives and false positives at high values of N_S. However, in order to avoid suspecting any normal flow, a low false positive rate is preferred even if it comes at the price of some worms being undetected. This problem can be partially overcome by increasing Δ_{TH}. But an alternative approach to threshold adjustment becomes necessary if sufficient worms do not enter the network in θ_T. One such alternative is to increase the L_S. The probability of a string's appearance in normal traffic decreases with an increase in its length. This decrease is, however, low for the presence of worm tokens in worm payloads. The effect of these two approaches is clearly illustrated in Fig. 3(b) where N_{FP} is reduced to zero but N_{TP} remains relatively unchanged even for high values of N_S.

Fig. 4 shows the values of N_{TP} and N_{FP} for different values of Δ_{TH}. In this case also, increasing the threshold Δ_{TH} alone is not sufficient to acquire a desired performance because it can not be bias enough to reduce only false positives. A typical of such case is depicted in Fig. 4(a). As shown in the figure, although it is possible to completely suppress false positives by raising Δ_{TH} beyond 6, doing so significantly reduces the detection rate. Poor detection implies that sufficient parts of worm payloads have not been cached. As indicated by Fig. 3, it is possible to solve this problem by increasing N_S. Fig. 4(b) potrays an example of such a solution. In this case, a significantly high number of true positives are obtained even at large values of Δ_{TH}. An increase in N_S obviously increases false positives. But, because worm payloads show higher resemblance with one another than normal flows, increases in occurrence frequencies of worm tokens are more than those of normal tokens. Hence, high false positive rates are limited to situations when the values of Δ_{TH} are low. In our experiments, a large number of worms were successfully detected, while maintaining minimal false positives, when $6 \leq \Delta_{TH} \leq 8, 15 \leq N_S \leq 20, L_S = 30$.

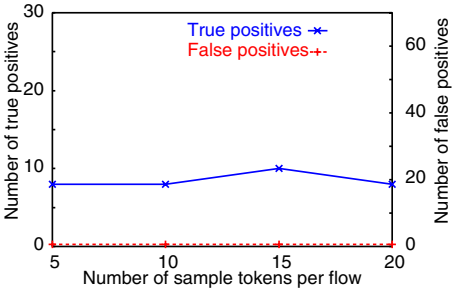
In our experiment, the average time taken by the ADU unit to analyze one minute's SUU filtrate was 1.87 seconds. This promising result indicates that the ADU analysis is practical for real-time detection of Internet worms.

C. Performance Evaluation of Generated Signature

Having evaluated the performance of a local manager, we now direct our focus to generating worm signatures at a metropolitan manager and to investigating the efficiency of the signature in detecting worms. A total of 21 flows for which

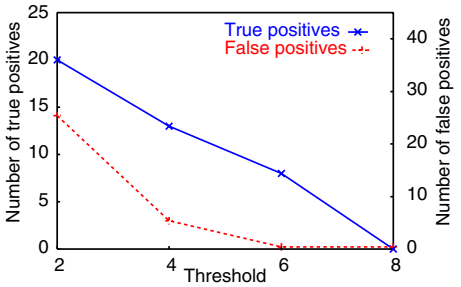


(a) $\Delta_{TH} = 4, L_S = 10$

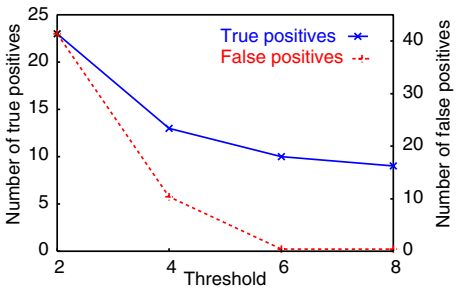


(b) $\Delta_{TH} = 6, L_S = 30$

Fig. 3. Number of true positives and false positives for different numbers of sample tokens extracted per flow ($\theta_T = 60$ min).



(a) $L_S = 30, N_S = 5$.



(b) $L_S = 30, N_S = 15$.

Fig. 4. Number of true positives and false positives for different repetitive occurrence thresholds ($\theta_T = 60$ min).

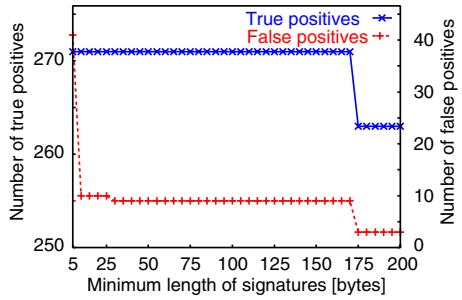
alerts were generated during different time stances of four different days were collected. These were used as suspicious flows sent by four local managers. After the clustering phase, the highest ranking cluster included seven flows. Tokens common to these flows and longer than the *Minimum Length of Signature-substrings* (L_{MIN}) are used as signature substrings. As L_{MIN} largely influences detection accuracy, this parameter is used to investigate the performance of these signature strings in detecting worms. We consider two scenarios. In Scenario 1, $ATL = 0$; an alert for a flow is generated if any signature substring is present in it. On the other hand, Scenario 2 is set under the condition $ATL = 0.5$; a flow is considered a worm if it contains at least half of the signature substrings.

Fig. 5(a) depicts N_{TP} and N_{FP} in case of Scenario 1. A 100 percent detection rate is achieved for L_{MIN} below 175 bytes. N_{FP}, on the other hand, shows a decreasing tendancy as L_{MIN} increases. The fact that the probability of a string's presence in normal flows decreases with respect to its length, and that the number of signature strings decreases with an increase in L_{MIN} are the reasons behind this phenomenon. In this scenario, best results are achieved when L_{MIN} is between 30 and 170. Under this condition, the detection rate is 100 percent while the number of false positives is nine. Fig. 5(b), on the other hand, plots N_{TP} and N_{FP} for different values of L_{MIN} in case of Scenario 2. Compared to Scenario 1, false positives have significantly reduced. This is most clearly observable when the value of L_{MIN} is between 5 and 15. It implies that the false positives generated for $L_{MIN} \leq 15$ in Scenario 1 were because of a small number of short substrings. In Scenario 2, a 100 percent detection rate with a low false positives (three) is achieved when L_{MIN} is set between 100 and 145. The fact that the majority of the signature substrings generated by the metropolitan manager are worm tokens, along with the requirement of half of signature strings needed to be present to generate an alarm, makes this method more accurate than that of Scenario 1. Given the results of both scenarios, it is clear that the propagation of worms in the concerned network could have been prevented, had the proposed system been implemented online.

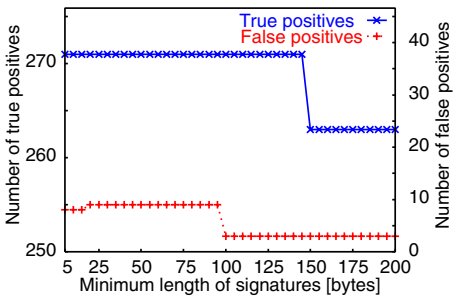
D. Discussion

Signatures generated by metropolitan managers can further be refined at the global manager. During a global propagation of a worm, the global manager is likely to receive similar alerts and signatures from different metropolitan managers. A refined signature can be generated by extracting common portions of the received signature substrings. Higher accuracy is thus possible. Besides, the refinement process also helps to reduce the time required during signature matching later at local managers because *fewer* and *shorter* substrings will be generated by the global manager.

Although it is possible to observe worm propagation from one MAN to another, such spread is more distinctly observable within MANs at the very early phase of a worm outbreak. Hence, monitoring geographically near networks is effective for an early detection of worms. While Indra [21] envisions a scenario where geographically distant peers warn each other of worm propagation, a high detection accuracy is expected from each participating peer in the network. In the proposed scheme, however, a slight error in detection at local managers is still bearable because only a few and most probable worm flows will be sorted for signature generation at metropolitan managers. The proposed scheme is applicable in architectures such as DOMINO [20] which comprises also hierarchically placed nodes apart from peer-to-peer components. Sink-nodes



(a) Scenario 1: $ATL = 0$



(b) Scenario 2: $ATL = 0.5$

Fig. 5. Number of true positives and false positives for different values of minimum length of signature strings for a real network data that contains 45,193 flows out of which 271 are worms.

present in DOMINO are not necessary in the proposed scheme because the proposed system deals solely with the traffic content.

Finally, it should be stressed out that the proposed scheme generates signature from worm payloads. Worms with encrypted payloads can not be subject to detection by our proposed scheme, unless local managers are provided with appropriate keys to decrypt the payload contents. Such an operation can be possible provided a prior request from hosts soliciting a certain level of safety certification.

V. Concluding Remarks

In this paper, a two-layer hierarchical architecture is proposed for an automatic detection of Internet worms over large-scale networks. In the proposed system, the network topology is divided into a number of metropolitan areas which are divided in turn into local networks. Local security managers gather inbound flows that transfer similar contents and send them to the metropolitan managers of their respective MANs. Metropolitan managers use cluster analysis to sort some worm flows, and generate worm signature substrings. They then send these signature substrings to the global manager which multicasts them to local managers to stop further propagation of the worm over the entire network. Performance of the proposed system is evaluated using real network traffic that contains traces of Beagle worm. The proposed scheme managed to successfully collect suspicious flows at local managers, and to sort worm flows at metropolitan managers. The signature developed by the metropolitan manager succeeded in detecting worms with high detection rates while maintaining low false positive rates.

Given the encouraging results of the proposed system, the authors are currently working on an on-line implementation of the system.

References

[1] N. Weaver, V. Paxson, S. Staniford, and R. Cunningham, *"A Taxonomy of Computer Worms,"* In Proc. of the First Workshop on Rapid Malcode (WORM 2003), Washington, DC, USA, 2003.

[2] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, *"Inside the Slammer Worm,"* IEEE Security and Privacy, Vol. 1, No. 4, Jul./Aug. 2003, pp. 33-39.

[3] C. Shannon, and D. Moore, *"The Spread of the Witty Worm,"* IEEE Security and Privacy, Vol. 2, No. 4, Jul./Aug. 2004, pp. 46-50.

[4] R. Scandariato, and J.C. Knight, *"The Design and Evaluation of a Defense System for Internet Worms,"* 23rd IEEE International Symposium on Reliable Distributed Systems, Florianopolis, Brazil, Oct. 2004.

[5] J.L.A. Yaneza, C. Mantes, and E. Avena, *"The Trend of Malware Today: Annual Virus Round-up and 2005 Forecast,"* Annual Virus Roundup, 2004, Trend Micro.

[6] D. Dagon, X. Qin, G. Gu, W. Lee, J. Grizzard, J. Levine, and H. Owen, *"HoneyStat: Local Worm Detection Using Honeypots,"* Recent Advances in Intrusion Detection (RAID), Sophia Antipolis, France, Sep. 2004.

[7] C. Kreibich, and J. Crowcroft, *"Honeycomb - Creating Intrusion Detection Signatures using Honeypots,"*, In Proc. of the 2nd Workshop on Hot Topics in Networks (HotNets-II), Cambridge, MA, USA, Nov. 2003.

[8] M.G. Schultz, E. Eskin, and E. Zadok, *"Data Mining Methods for Detection of New Malicious Executables,"* IEEE Symposium on Security and Privacy, Oakland, CA,USA, May 2001.

[9] D. Evans, and D. Larochelle, *"Improving Security Using Extensible Lightweight Static Analysis,"* IEEE Software vol. 19, no. 1, Jan./Feb. 2002, pp. 42-51.

[10] J. Newsome, and D. Song, *"Dynamic Taint Analysis for Automatic Detection, Analysis, and Signature Generation of Exploits on Commodity Software,"* In Proc. of the 12th Annual Network and Distributed System Security Symposium, Alexandria, VA, USA, Feb. 2005.

[11] C. Cowan, S. Beattie, J. Johansen, and P. Wagle, *"PointGuard: Protecting Pointers from Buffer Overflow Vulnerabilities,"* In Proc. of 12th USENIX Security Symposium, Washington, DC, USA, Aug. 2003.

[12] X. Chen, and J. Heidemann, *"Detecting Early Worm Propagation through Packet Matching,"* Technical Report ISI-TR-2004-585, USC/Information Sciences Institute, Feb. 2004.

[13] M. Bhattacharya, S. Hershkop, and E. Eskin, *"MET: An Experimental System for Malicious Email Tracking,"* In Proc. of the 2002 New Security Paradigms Workshop, Virginia Beach, VA, USA, Sep. 2002.

[14] C.C. Zou, W. Gong, and D. Towsley, *"Feedback Email Worm Defense System for Enterprise Networks,"* University of Massachusetts, Technical Report TR-04-CSE-05, Apr. 2004.

[15] P. Akritidis, K. Anagnostakis, and E.P. Markatos, *"Efficient Content-Based Detection of Zero-Day Worms,"* In Proc. of the International Conference on Communications (ICC 2005), Seoul, Korea, May 2005.

[16] H. Kim and B. Karp, *"Autograph: Toward automated, distributed worm signature detection,"* In Proc. of 13th USENIX Security Symposium, San Diego, CA, Aug. 2004.

[17] S. Singh, C. Estan, G. Varghese, and S. Savage, *"The Early-Bird system for real-time detection of unknown worms,"* Technical Report CS2003-0761, University of California, San Diego, Aug. 2003.

[18] J. Newsome, B. Karp, D. Song, *"Polygraph: automatically generating signatures for polymorphic worms,"* IEEE Symposium on Security and Privacy, Oakland, CA, USA, May 2005.

[19] S. Chen, Y. Tang, *"Slowing Down Internet Worms,"* In Proc. of 24th International Conference on Distributed Computing Systems (ICDCS'04), Tokyo, Japan, Mar. 2004.

[20] V. Yegneswaran, P. Barford, and S. Jha, *"Global Intrusion Detection in the DOMINO Overlay System,"* 11th Annual Network and Distributed System Security Symposium, San Diego, CA, USA, Feb. 2004.

[21] R. Janakiraman, M. Waldvogel, and Q. Zhang, *"Indra: A peer-to-peer approach to network intrusion detection and prevention,"* In Proc. of 2003 IEEE WETICE Workshop on Enterprize Security, Linz, Austria, Jun. 2003.

[22] K. Simkhada, H. Tsunoda, Y. Waizumi, and Y. Nemoto, *"Differencing Worm Flows and Normal Flows for Automatic Generation of Worm Signatures,"* In Proc. of the Seventh IEEE International Symposium on Multimedia (ISM2005), Irvine, CA, USA, Dec. 2005, pp. 680-685.