Journal of Computer Science 7 (11): 1724-1728, 2011
ISSN 1549-3636

Behaviour Based Worm Detection and Signature Automation

[1]Mohammed Anbar, [1]Selvakumar Manickam, [2]Al-Samarraie Hosam,
[1]Kok-Soon Chai, [1]Mohmoud Baklizi and [1]Ammar Almomani
[1]National Advanced IPv6 Centre of Excellence,
[2]Centre for IT and Multimedia,
University Sains Malaysia, Penang, Malaysia

Abstract: Problem statement: A worm is a malicious piece of code that self-propagates, often via network connections, to exploit security flaws in computers connected through the network. In general, worms do not need any human intervention to propagate and are considered a real threat to network assets and the properties of organizations. An Intrusion Detection Systems (IDSs) are employed to detect the presence of the worms in the network. **Approach:** This study proposed a new behaviour-based worm detection and signature automation approach that consists of scanning characteristics to find vulnerable hosts and indicate the correlation between an infected host and potential destination hosts. **Results:** This approach can be distinguish between network scanning (random and sequential TCP and UDP worm scanning) triggered by infected and non-infected hosts. In addition, the ability to detect the worms based on its behaviours. **Conclusion:** Identifying network worms at an early stage can increase the protection of network services and vulnerable hosts.

Key words: Network scanning, worm detection, Intrusion Detection Systems (IDSs), Artificial Neural Networks (ANNs), Destination-Source Correlation (DSC)

INTRODUCTION

Nowadays, many organizations share information through a network and make the data available and accessible via the Internet. Due to this fact, there has been a significant increase in daily transactions that are made on the internet; these are vulnerable to significant threats such as unauthorized access or theft of private information. This dependency on the internet makes network assets and information on the network a valuable target for attackers and hackers. One of the most prevalent threats to networks is network worms because these can spread without human intervention. Once a worm infects any host in a network, it will have huge destructive effects over the network topologies and resources. Many Intrusion Detection Systems (IDSs) are deployed in the edge router and default gateway to detect worms before they infect the network. However, many network worms can go through IDSs and successfully infect the network, especially zero-day worms (i.e., worms with signatures that do not exist in the IDS signature database).

The severity of computer worms has grabbed researchers' attention in the last few years. Many approaches have been proposed for behaviour-based worm detection and signature automation; some of these approaches are based on Artificial Neural Networks (ANNs) others are based on the connection failures that occur in the network.

A scholar by Stopel *et al*. (2006) proposed an approach for detecting worm-infected hosts that is based on an Artificial Neural Network (ANN). This approach measures properties of the infected host such as Central Processing Unit (CPU) and memory usage; these computer measurements have high dimensionality, which makes the training process time very long. Thus, feature selection techniques are employed to reduce the dimensionality; some techniques are as follows: (1) finding the relation between the inputs and hidden neuron's relative variance; (2) the Fisher score ranking; and (3) the gain ratio filter. The outputs of these techniques are features that impact the behaviour of computers which are infected by worms. These techniques evaluate each technique by pre-processing the dataset and training the ANN model with the pre-processed data based on the training dataset. The ability of the model to detect the presence of a new computer worm is then evaluated, particularly during heavy user activity on the infected computers. However, many worms (especially zero day worms) bypass the IDSs because its signatures does not exist in signatures database (Singh *et al*., 2005).

Corresponding Author: Mohammed Anbar, National Advanced IPv6 Centre of Excellence, University Sains Malaysia, Penang, Malaysia

In contrast, Moskovitch *et al.* (2008) proposed an approach based on computer behaviour that uses Artificial Neural Networks (ANNs) to classify the computer as infected or not. To validate his assumption, he infected a computer with five different worms having different behaviours and ran several different applications (e.g., MSN, Windows Media Player and Microsoft Word). He monitored and logged 323 features to create eight datasets, each containing separate monitored samples of each of the five injected worms and samples of a normal computer's behaviour without any injected worms. The ANN approaches are computationally advantageous when real-time computation is needed and have the potential to detect previously unknown worms with a high level of accuracy. In addition, ANN can reduce the feature dimensionality. The two biggest shortcomings of ANN techniques are the (1) training period (they take a long time to train) and (2) the involvement problem (any changes in the target environment affects the training dataset).

MATERIALS ADN METHODS

The proposed approach, which is named behaviour-Based Worm Detection and Signature Automation (BBWDSA), is based on the assumption that the first step performed by network worms is to scan a network for vulnerable hosts and services; once a vulnerable host is found, the malicious code will be transferred from the sender to the destination. The packets used to transfer malicious code from the sender to the destination have specific and noticeable traffic.

Once the malicious code infects the target host, the target host scans the network to find vulnerable services that have been infected. Figure 1 shows the architecture of the BBWDSA approach.

BBWDSA consists of three sub-approaches:

- The network scanning approach aims to detect TCP and UDP random and sequential scanning and consists of three sub-modules: (1) a filtering module, (2) traffic statistical analyzer module and (3) cross-relation module
- The network worm's correlation approach aims to detect Destination Port Correlation (DPC) behaviour for detected scanning IPs in the network and consists of two sub-modules: (1) the Destination Port Correlation Based Worm Detection module (DPCBWD) and (2) the alert module

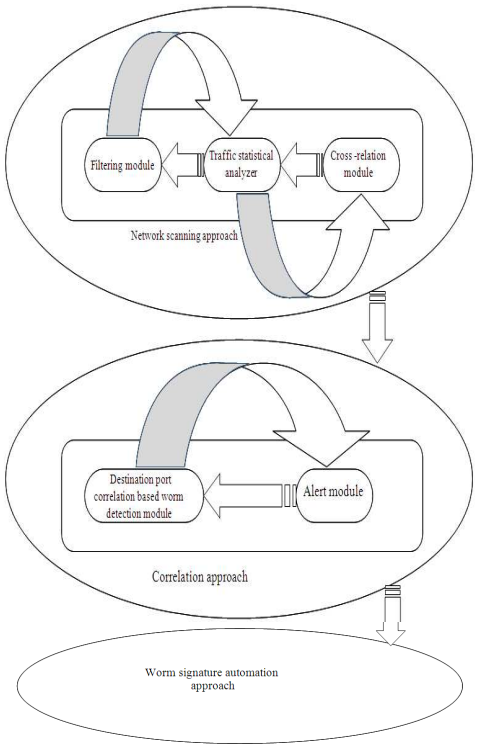

Fig. 1: Architecture of the BBWDSA approach

- The worm signature automation approach aims to generate a behaviour signature for detected worms

Network scanning approach: Network scanning is used to identify active hosts, services, operating systems and applications running on each computer system within the targeted network; it is considered the first step for an attacker to gain access to the network.

Network worm's correlation approach: Worm behaviour is usually repetitious and predictable, which makes it possible to be detected. As defined in (Gu *et al*., 2004) worm behaviour can be predicted by correlating an incoming connection on a given port with a subsequent ongoing infection at that port; this behaviour is called Destination-Source Correlation (DSC).

The start point for this approach begins after the network scanning approach sends out the scanning IPs. The correlation approach consists of two sub-modules (i.e., Destination Port Correlation Based Worm Detection **(**DPCBWD**)** and alert modules). Figure 2 shows the correlation approach flow chart.

Destination port correlation based worm detection module (DPCBWD): Many approaches have been

proposed to detect network worms. One is based on connection failure; this frequently occurs in a network being scanned. Another is based on using an ANN to detect network worms. The drawback of these approaches is a high false positive rate because these approaches do not consider all abnormal behaviours for network worms which can clearly appear in the target finding (e.g., network scanning) and propagation (e.g., source and destination port correlation) phases of the network worm life cycle. The DPCBWD module was proposed to detect destination port correlation (DPC) between the scanning IPs which detected by scanning approach and source IPs. The following is an example of destination port correlation. Suppose that a host receives a packet on port i and then starts sending packets destined for port i. If the number of sending packets destined for port i to different destination hosts exceeds the predefined threshold, the host becomes suspicious. Figure 3 shows DPC behaviour.

Suppose that 192.168.1.2 is detected as a scanner IP (as detected by the scanning approach). As shown in Fig. 3, host 192.168.1.13 sends out packets targeting port 25 to other hosts (192.168.1.13, 192.168.1.30, 192.168.1.15, 192.168.1.77, 192.168.1.7 and 192.168.1.2). Since host 192.168.1.2 sends out the received packet to other hosts with the same port number (25), this means that 192.168.1.2 is a vulnerable host that exhibits DPC behaviour. On the other hand, hosts 192.168.1.13, 192.168.1.30, 192.168.1.15, 192.168.1.77 and 192.168.1.7 do not send any packets targeting port 25, which means that these hosts do not exhibit DPC behaviour. Thus, we can conclude that 192.168.1.2 is an infected IP since it exhibits scanning and DPC behaviours.

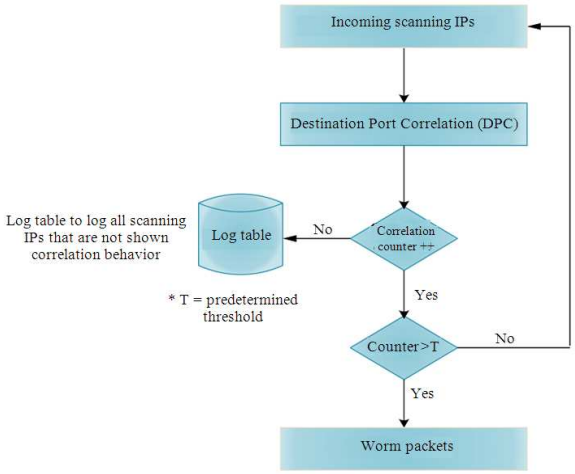

Fig. 2: The correlation approach flow charts

As each worm performs a scan but the opposite is not true, the DPC behaviour in addition to scanning behaviour is considered to detect the presence of worms in the network rather than considering the scanning behaviour only to increase the accuracy of worm detection.

Since existing malicious codes share the scanning activities with network worms, the scanning log Table is created to log all IPs that perform network scanning but do not show DPC behaviour for further analysis by a network administrator.

Meanwhile, some applications exhibit DPC behaviours but are not worms; to overcome this problem, port DB is used to log all applications ports that exhibit DPC-like behaviour. An example of this type of application is Gnutella. Gnutella may receive TCP/6346 traffic as well as send data to other clients through TCP/6346 elsewhere. The Gnutella network is a peer-to-peer (P2P) network, which allows users on different networks to share files. However, each user still must connect to an 'ultrapeer', which is a server that lists files shared by connected users. This makes it possible to search for files across hundreds or even thousands of other computers connected to the network. Gnutella clients include Acquisition for Mac and BearShare and Morpheus for Windows. By considering the applications that exhibit DPC-like behaviour, the occurrence of false positives can be reduced. After worm packets are received from the DPC, the destination port for each packet is extracted and compared with existing ports. If a match exists, the worm packet is ignored; otherwise, the worm packet is forwarded to the alert module.

Alert module: This module is responsible for generating alerts for detected worms and scanner IPs. The generated alerts are presented as reports. Table 1 shows an example of alert report information for detected worms Table 2 shows alert report information for scanning IPs.

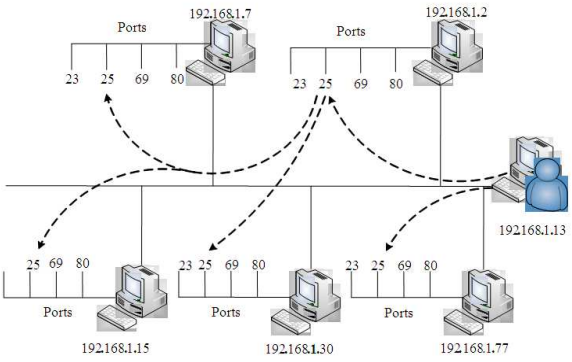

Fig. 3: DPC Behaviour

Table 1: Example of alert report information for detected worms

Source IP	Destination	port	Worm type severity (%)
10.20.200.160	80	TCP	98
10.20.200.140	90	UDP	56
10.20.200.200	80	TCP	40
10.20.200.120	90	TCP	10

Table 2: Example of alert report information for network scanning

Source IP	Destination IP	Destination port	Scanning type
10.20.200.160	10.20.200.98	1434	UDP Random scanning
10.20.200.160	10.20.200.98	1434	UDP Sequential scanning
10.20.200.70	10.20.200.172	25	TCP Sequential scanning
10.20.200.30	10.20.200.22	80	TCP Random scanning

Table 3: Example HS table

Port	Protocol	Services
7	TCP	Echo
21	TCP	FTP
22	TCP	SSH
53	UDP	DNS
67,68	UDP	DHCP
80	TCP	HTTP
135,1025	TCP	DCOM
445	TCP	NetBIOS
445	UDP	NetBIOS
5900	TCP	VNC
5000	TCP	UPNP

Calculating the severity percentage and scanning rate: The alert module consists of two tables: High Severity (HS) and Low Severity (LS) port table. Each table consists of three fields (port, protocol and service). The ports are classified based on DShield. DShield provides reports about most ports attacked per target and source. The ports in HS have one configurable weight as well as the ports in LS. Table 3 shows the sample ports for HS.

The following equation is used to calculate the severity percentage:

$$\text{Severity percentage for i} = \left(\frac{\text{Total destination address of i}}{\text{Total destination address for all in fected ips}} w + \right) \times 100\%$$

where, i is the infected IP and w is the weight of the destination port for the detected worm.

The worms and network scanning have destructive effects on the network resources and topology. Therefore, detecting worms and network scanning at an early stage provides the network administrator with the chance to take early action before the network machines are compromised. The alerts provide the network administrator with information about the worm severity and infected machine as well as scanning behaviour of the infected network. Such information is useful for facilitating suitable actions and correct decisions, such as installing a firewall and anti-virus software or updating the existing IDS.

Signature automation approach: The worm signature is a specific string that exists in the packet payload. Signature-based IDSs such as Snort (Snort) compare this string with existing signatures in the database. If there is a match, the worm can be detected. One of the biggest drawbacks for signature-based IDS is that they cannot detect zero-day worms (i.e., the worm's signature does not exist in the database). Meanwhile, some NIDS exist that check the content of network traffic; these include AutoGraph (Kim and Karp, 2004), EarlyBird (Sen *et al*., 2004), Anagram (Wang *et al*., 2010) and the LESG (Li *et al*., 2006) polymorphic worm (its signature can be changed each time it is sent to a vulnerable host).

The signature automation approach aims to generate a behaviour signature for detected worms; the entry point for this approach is the network worm that is received from the network worm's correlation approach, which takes the thresholds (scanning and DPCBWD approaches), used for the detected worm and automates the behaviour signature for the detected worm. The general behaviour signature rule is as the follows:

- <IP, Protocol type, threshold, time window >And
- <IP, DSP, threshold, time window >

where, IP is the infected host, protocol type = {ICMP-T3-1or ICMP-T3-3or TCP-REST or TCP-SYN}, the threshold is a predefined value for {ICMP-T3-1, ICMP-T3-1, TCP-REST, TCP-SYN and TCP-SYN/ACK}, the time window is a specific time value and DSP is a Boolean (true or false) flag for if the IP exhibits DSP behaviour. The generated rules are used to detect TCP and UDP worms.

RESULTS AND DISCUSSION

The key points that distinguish our proposed approach from many other similar works are as follows:

- No need for prior knowledge of network worms since it is behaviour-based
- The proposed method has a new approach to detect random and sequential TCP and UDP worm scanning
- The accuracy of network worm detection compared to other similar approaches is better because the approach is based on two worm detection behaviours: network scanning (by employing a

new approach for scanning detection) and correlation between the source and destination host
- The approach can distinguish between network scanning (random and sequential TCP and UDP worm scanning) triggered by infected and non-infected hosts
- The signature automation approach generates a behaviour signature that is not based on the packet payload (to extract the worm signature)

CONCLUSION

Network worms are self-propagating malicious codes with destructive effects on network resources and topologies. Network worm detection is a challenging problem; in this study, we propose a new approach for worm detection and signature automation, which we named behaviour-based worm and signature automation that is based on three common network behaviours. In the near future, we plan to further improve the accuracy and efficiency of our proposed detection approach and develop and implement a general system for the detection of network worms.

REFERENCES

Gu, G., M. Sharif, X. Qin, D. Dagon, W. Lee and G. Riley, 2004. Worm detection, early warning and response based on local victim information. Proceedings of the 20th Annual Computer Security Applications Conference, Dec. 6-10, IEEE Xplore Press, pp: 136-145. DOI: 10.1109/CSAC.2004.51

Kim, H.A. and B. Karp, 2004. Autograph: Toward automated, distributed worm signature detection. Proceedings of the 13th conference on USENIX Security Symposium (SSYM'04), USENIX Association Berkeley, CA, USA., pp: 19-19.

Li, Z., M. Sanghi, Y. Chen, M. Y. Kao and B. Chavez, 2006. Hamsa: Fast signature generation for zero-day polymorphicworms with provable attack resilience. Proceedings of the IEEE Symposium on Security and Privacy, May 21-24, Berkeley/Oakland, California, pp: 32-47.

Moskovitch, R., Y. Elovici and L. Rokach, 2008. Detection of unknown computer worms based on behavioral classification of the host. Comput. Stat. Data Anal., 52: 4544-4566. DOI: 10.1016/j.csda.2008.01.028

Sen, S., O. Spatscheck and D. Wang, 2004. Accurate, scalable in-network identification of p2p traffic using application signatures. Proceedings of the 13th international conference on World Wide Web, May 17-22, New York, USA., pp: 512-521. DOI: 10.1145/988672.988742

Singh, U.K., A.K. Ramani, N.S. Chaudhari and V. Gupta, 2005. Increasing effectiveness of ids to improve security in intranet. Am. J. Applied Sci., 2: 1032-1035. DOI: 10.3844/ajassp.2005.1032.1035

Stopel, D., Z. Boger, R. Moskovitch, Y. Shahar and Y. Elovici, 2006. Improving worm detection with artificial neural networks through feature selection and temporal analysis techniques. Int. J. Applied Math. Comput. Sci., 1: 34-40.

Wang, Y., Y. Xiang and S. Z. Yu, 2010. An automatic application signature construction system for unknown traffic. Concurrency Comput.: Practice Experience, 22: 1927-1944. DOI: 10.1002/cpe.1603