Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 7, Issue, 03, pp.13223-13230, March, 2015

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

ISSN: 0975-833X

RESEARCH ARTICLE

MALICIOUS ACTIVITIES AT DEFENSE AND ATTACK STAGE - MALWARE'S

***Ranu Lal Chouhan and Govind Singh Tanwar**

Department of Computer Science and Engineering, Govt. Engineering College, Bikaner

ARTICLE INFO

Article History:
Received 28th December, 2014
Received in revised form
15th January, 2015
Accepted 20th February, 2015
Published online 17th March, 2015

Key words:
Malware, Worm, Propagation,
Attack, Defense,
Lifecycle, Anomaly.

ABSTRACT

Malware are malicious software, capable of replicating themselves in network. They are designed to disrespect user choice of computing in network system. This paper presents current state of art of malware attacks and defense life cycle. We explored the characteristics of malware. We have presented the malware attack life-cycle. We have discussed different kind of strategies employed for target acquisition, transferring, and activation. Different kinds of malware employ different strategies in their attack life cycle. Therefore researchers have worked in different directions to cover the diversity of malware attacks. We have presented current state of art in malware defense. The purpose of this review is to point out the strength and challenges of contemporary malware defense mechanisms. We believe that this study will help security researchers in choosing appropriate mechanism for malware defense. The pros and cons of defense strategies will also helps in building more robust defense techniques in future.

INTRODUCTION

Definition of Malware

Malwares are the malicious codes or malicious software, which take unauthorized entry into a system or network of systems with or without human intervention. In fact the term malware was originated from *Mal*icious Soft*ware*. Malwares are broadly classified into two categories: One that need host programs to propagate. Most common malwares in this category are virus, Trojan horse, logic bomb, trapdoor etc. The other types of malware exist or propagate independently without help of any application program, utility software or system software. Worms, zombies are examples of such malware. Among them virus and worms are self replicating.

What constitute a Malware?

The scope of our paper is "defense against malware". For this reason it is very important to understand what precisely makes a malware. Unfortunately, after reviewing the primary sources, it has been observed that there is very little or no consensus on definition of different malware. However we have chosen to include the following aspects of definition of malware.

Malicious Program

It is generally agreed that, be it worms, virus or any other malware, all are malicious codes. There exist some "good worms" which are primarily used for system repairing and maintenance.

**Corresponding author: Ranu Lal Chouhan,*
Department of Computer Science and Engineering Govt. Engineering College Bikaner.

These programs are not termed as malware. Malware is defined as "Programs that are intentionally designed to perform some unauthorized (and often harmful or undesirable) act." (Network Associates, 2003)

Network Propagation

Another proven fact is that, malware propagates over a network. They actively use different network interfaces, such as emails, shared network connections or any other network interface (Mohsen Damshenas *et al.,* 2013) that connects one or more homogeneous or heterogeneous network. The worm propagation may be epidemic (Mishra and Saini, 2007).

User Interaction

There are varied degrees of user intervention or interaction is required for malware propagation. Viruses required human intervention such as transferring file from one system to another system using removable disk or executing a particular program. Worms are said to propagate without user interaction. Once a system is infected by worms it can spread itself with little or no human intervention. But another class of thought classifies worms into two categories: One that requires user interaction such as downloading or opening a malicious attachment from emails etc. and other which require no user intervention to propagate in network.

Self Replication

Some malware replicate themselves to propagate in network, similar to biological pathogens. Virus, worms are example of

such malware. F-secure virus glossary defined a virus as "a computer program that replicates by attaching itself to another object" (F-Secure, 2003) and a worm is "a computer program that replicates independently by sending itself to other systems" (F-Secure, 2003). There is a class of thought which does not make this distinction. For them any malicious software that replicates in order to spread themselves in network are termed either as virus or worms. The nature and characteristics of malware are fast changing. Therefore any classification of malware may soon prove to be incomplete. There may also be overlapping characteristics among different categories of malware.

In this paper, we have used the term worms and virus interchangeably without much distinction. We focus our defense on dynamic malware which replicates in order to propagate in network. We include virus, worm or other malicious code in our definition of malware which propagates independently or by attaching itself with some objects with or without human intervention. We adopt the following definition of malware in this paper: "A malware is a program that self propagates across a network exploiting security or policy flaws in widely-used services."(Weaver *et al.,* 2003)

Worm attack life-cycle

Steps: Target discovery, malware transferring, malicious code activation and infection.

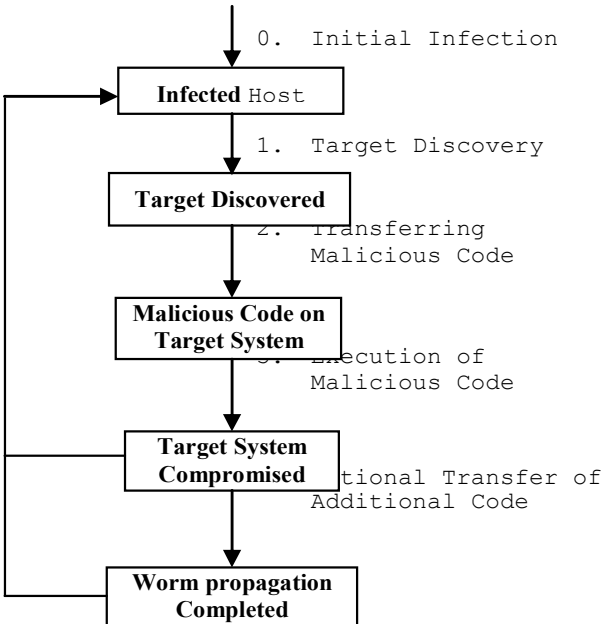

Fig. 1. Worm attack life cycle

Worm Target Discovery

Target discovery is the first step of operation by a worm to find potential victims. Worms use several innovative techniques to discover potential victims. The speed of worm propagation largely depends on the kind of scan employed by the worms. The most common scanning techniques are blind scanning, hit-list scanning, topological scanning, and passive search techniques. A combination of such approaches may also be employed by the worm for target acquiring. It is very important to know about these algorithms to successfully defend them. If defense can block any such scheme the entire class of worm, which employ it, can be prevented from transferring into another system.

Blind scanning

In this scheme worm finds the target without any prior knowledge of victim. There are three categories of such scanning: Random scan (worm tries to acquire victim by scanning a block of address in random manner), Sequential scan (worm scan through randomly generated ordered list of addresses) and Permutation Scan (Stuart Staniford *et al.,* 2002) (worm uses distributed coordination for scanning the entire address space.). Because of the simplistic nature of blind scanning, this method is adopted by many autonomous worms such as Code Red (eEye Digital Security, ?) Nimda (F-Secure Virus Descriptions, 2001), Slammer (Moore *et al.*, 2003). Sequential scan and random scan spread worms relatively slowly, but coupled with automatic activation it may spread faster. However such scans are not very accurate. As there is no prior knowledge of victims address, missing rate may also be very high. Scanning is highly anomalous behavior. Different anomaly based detection algorithm can easily detect such worm (Jiankun Hu and Xinghuo Yu, 2009; Snort, ?; Bolzoni and Hartel, 2006). There are several improved and optimized version of scanning techniques, which are difficult to detect. These techniques are discussed below.

Subnet Scanning

This is an improved version of blind scan. In this scanning technique, worm scan for the vulnerable host in the same subnet, instead of scanning entire internet, collecting information from current victim host. Hit rate ratio in this method may be relatively high. Once the worm enters into an organization gateway, it can spread into all vulnerable machines inside the gateway since security against malware may not be that robust as desired. The speed of spread of scanning worm depends on following factors: Scanning optimization technique, density of vulnerable population, design of scan routine, range of address space (IPV4 vs. IPV6). In IPV4 (total 232 IP address in entire IPV4 space) it is quite easy to scan entire address span but when IPV4 is upgraded to IPV6, it exponentially increase the address space (In case of IPV6 total 264 IP address in single subnet). As a result, it becomes very difficult to scan IPV6 space (Li *et al.,* 2008), thereby preventing the spread of scanning worms dramatically.

Hit list Scan

A pre-generated list of vulnerable host is acquired by the attacker in this technique. Although it is relatively easy to create small list of vulnerable hosts through public sources or open access points, creating and carrying a comprehensive hit list may be difficult for several reasons. Compare to blind scan, hit-list scan is more accurate and fast. However a hit-list may be outdated because of dynamic nature of internet address space. It is difficult for anomaly based network intrusion

detection (NIDS) to detect hit-list attack, because the list is pre-generated and miss rate is low.

Topological Scanning

There are instances when some application in certain host contains information about other vulnerable hosts in networks. Topological worms gain the knowledge of local communication topology using this target list. A very popular example of such worm is email worms, which uses the address book of the victim to initiate attacks. Morris worm (Mark Eichin and Jon Rochlis, 1989) is also an example of topological worm which uses /etc/host and other network resources to discover local communication topology. Topological worms can be very fast because they have the local information and need to connect only these hosts. Moreover topological worms are hard to detect as the accuracy of attack is high and local attack behavior is more like normal traffic. However global anomaly may be detected for topological worms.

Passive Technique

Passive worm does not scan the network; instead they wait for vulnerable host to connect them. The passive worm replies the host with a copy of it. Passive worms are in generally slow because they need to wait for potential victim to interact with them, but they are very hard to detect with anomaly band detection. Gnuman (Gamespy, ?), Contagion (Stuart Staniford *et al.,* 2002), CRClean (Markus Kern, ?) are few examples of passive worm.

Worm Scanning Constraint

Based on constraint of scanning worms are of divided into two categories: Latency limited and Bandwidth Limited. TCP worms are usually latency Limited whereas UDP worms are bandwidth limited.

Bandwidth Limited

The bandwidth worms are connection less. UDP worm are seen to be bandwidth limited. They do not require establishing a connection before spreading into next vulnerable hosts. They don't have to wait for acknowledgement packet from the host before initiating connections. Such worms uses self-carried scheme for propagation and send a copy of it along with first packet sent to victim for infection. As they do not require to wait for any response, they are very fast; usually at the order of hundred mbps or more. These worms are bandwidth limited because the data or packet generated by them sometimes exceeds the bandwidth of the network. Slammer/Sapphire is example bandwidth limited worm.

Latency Limited

TCP worms such as Code-red-I and code-red-II requires to establish a connection before spreading. Such worm send TCP/SYN packet to the targeted host and then waits for a response. Either it will receive a SYN/ACK packet or a timeout from the host. The worm can't take any action during the waiting time also known as latency. Compared to bandwidth constrained worm latency limited worm require an additional round trip time and 40 bytes of overhead in each round to establish the connection.

Malware transfer mechanism

In last section we have seen different technologies adopted by different worms to find the vulnerable hosts. Once the potential victim is discovered, worm can use different propagation scheme to spread themselves. Different Propagation Schemes are:

Self carried

This scheme is very straightforward. It transmits its payload by itself when infecting other victims. Self activating worm or topological worm usually adopts this mode of propagation. Even some passive worm such as CRClean (Markus Kern, ?) employ self carried scheme for worm transmission.

Second Channel

After finding the next victim worm is transferred into it but it use a secondary channel to download the harmful code from internet or infecting machines, using TFTP, RPC or some other applications. Blaster (Jason V Miller *et al.,* ?) is an example of worm which uses second channel propagation scheme to transmit its payload into victim machine.

Embedded

A more deceptive worm may append its payload or replace a legitimate message to spread itself into other machine. Embedded propagation scheme is very effective for the worm, because it rarely generates anomalous behavior. Contagion worm (Stuart Staniford *et al.,* 2002), a passive worm uses embedded propagation strategy (Li *et al.,* 2008)

Worm Payload Activation

A worm can be activated in several ways. Some worms are activated by human intervention; some are activated by schedule process execution. Some worms are self activated. The activation mechanism has large impact on the worm propagation speed.

Self Activation

On a network based system, there are certain services which are always running such as IIS Web Servers. Few worms attach themselves to these running programs or with the libraries that these programs use. Whenever a vulnerable service is discovered the worms get activated along with the running program. These worms are transmitted to the next vulnerable services. Code Red –I (eEye Digital Security, ?) is an example of worm which is self activated along with IIS server. This is fastest mode of activation.

Human Activation

Certain worms require as local user to intervene in executing its payload. Some social engineering approaches are used to

lure the local user to execute its code. An email attachment containing message sharing urgency by someone you know, so that such attachment will be opened and thereby executed the malcode along with it. Melissa worm (CERT, 1999) is an example of this type. I love you worm (CERT, 2000) attract the user with messages appear to be from loved ones. So that user opens it and activates it in the process.

Human Activity-based activation

There are certain worms which are activated by human activity, not necessarily related to direct execution of worm. When a user reset or configures a system these worms get activated along with executed code. User may remotely execute an infected file thereby spreading the infection. Nimda ("F-Secure Virus Descriptions, 2001) is an example of such worms which get activated when user login into system, thereby executing login script or when the machine is reset or reconfigured. Worm can write their payload into user memory when such activity is triggered.

Scheduled Process Activation

These are some worms which take the opportunity to get activated when some scheduled process is running. Many operating systems and application automatically download and install some updater program in user machine. If such program runs without authentication, few worms may creep into the system with these updater programs. Apart from auto-update program some system periodically run auto-backup and some other network softwares. These softwares may have vulnerabilities which are easily exploited by the worms.

Worm payload format

The actual code of the worm is termed as payload. The payload varies in their complexity from simple straightforward worm code to much complex payload. As the complexity of worm payload increases, detection becomes difficult. The worm with simple straightforward payload is called monomorphic worm. There can be variations of monomorphic worms by simply inserting some garbage data into the worm payload. Signature based approach still detect such worm as the signature remain the same.

Polymorphic worms change their payload dynamically on every infection attempt. Each instance look different but functionality remains same. There are several ways of achieving polymorphism. One way is to change the variant code in each instance. As the worms change their appearances dynamically it is hard to detect by traditional signature based approach.

Metamorphic worms not only change their appearances but also change their behavior (functionality) in every infection instance. Metamorphic worms adopt different code obfuscation technique to achieve the goal.

Malware defense strategy

Worm defense is not a single isolated activity. In order to systematically design the worm defense mechanism we observe that there are two fundamental approaches towards this problem. Either protect the vulnerable host from incoming malware attack or contain a local infection from sending outgoing attacks to spread the worm. There are many sub-problem of this problem. A worm is after all a malicious piece of code which exploits vulnerability in a system remotely. So the first step of worm defense strategy is to prevent the vulnerabilities to occur in operating system, application software and other network application software installed in a system. Although several technologies are in place to mitigate vulnerabilities, it is unfair to expect that all vulnerabilities will be eliminated. Assuming that unmitigated vulnerabilities will be there and they will be exploited by attackers, the next step of defense is to detect those attacks as early as possible of their occurrence. Detecting known attacks is fairly easy but the real challenge lies in detecting previously unknown worms. Therefore we should be ready to mitigate and respond to the attacks that evaded detection. In this section we will discuss different defense strategies with their strength and limitations.

Malware Prevention

Malware prevention is the first step of defense strategy. There are four primary elements of malware prevention: Policy, awareness, vulnerability mitigation and threat mitigation.

Malware prevention Policy

To maintain a healthy cyberspace every organization must have a clear policy addressing malware prevention. This policy goes a long way in protection digital asset of the organization. Malware prevention policy should cover internal as well as external (those who are working on organization's network space remotely, business partner, mobile devices etc.). Most common malware prevention policies expected to be adopted by an organizations are:

- Scanning of external media for malware by approved antimalware before use it in organization network.
- Email attachment or compressed file (zip file) to be saved in local drive first. Scan them before transferring into another machine or network drive.
- Restricting installation of unnecessary software; they may be potential carrier of the malware.
- Installation / updation of operating system and other software in local machine under strict privilege of administrative access or administrative monitoring.
- Access to external network including internet only through organization approved secured mechanism.
- Policy of procuring software, antivirus, operating system, spyware and other internet security tool as per threat perception of the organization.
- Restricting firewall and router configuration change only under strict administrative privileges.
- Restricting use of mobile devices in organization network domain.
- Logging into organization network with authentication so that a log of user activity can be traced at administrative level.

Malware prevention awareness

Cyber health education and awareness are of great importance in protecting and maintaining a healthy cyber space. Individuals are the main consumers of cyber health education and awareness programme, which promote safety and secure cyberspace environment. Communities, service providers, software and hardware vendors, government and other organizations are main providers of cyberspace education and awareness. Users should be aware, how malware enter into systems and spread in networks, risk associated with malware incidents, technical limitations of handling all malware incidents and importance of taking preventive measures. Users should also be aware of policies and procedures of organizations regarding malware protection. Towards the safe cyberspace behavior some common recommendations are:

- Take caution to open following certain file extensions such as .exe, .bat, .vbs, .com etc.
- Taking caution in opening the suspicious e-mails, attachments etc even if they seem to originate from known sources.
- Restricting visit and download from untrusted website. Advisory of some commercial anti-malware may be taken into consideration in this regards.
- Not disabling additional security features such as firewall, antivirus, auto update to security and other software etc.

Vulnerability Mitigation

There can be several vulnerabilities in operating system, utility software, application software and web services. Malware tries to exploit these vulnerabilities to get access of the system. It is very important to mitigate the vulnerabilities present in the system in order to avoid malware incidents. There are several ways we can mitigate vulnerabilities. Effective among them are discussed in following section.

Patch management

Patch management is systematically detecting and eliminating known software (including operating systems) vulnerabilities in the system to reduce the incident of vulnerability exploitation by the malware. The process can be manual by experts or automatic. There are several vendors available which provide patch management services. Automated patch management process may be administered from a centralized console so that IT administrator can select, install and manage patches in their client machines. Many patch management service provider scan the client machines in periodic interval for any potential vulnerabilities. IT administrator can also schedule patching services in their convenience. But there several challenge of patch management process.

- As the systems are increasingly become more and more complex the numbers of vulnerable patches are increasing, making it difficult even for automated patch management vendor to eliminate the security holes fully.
- Not all users are aware to keep an eye on Patch release. For example patch for SQL Slammer worm was discovered 6 months before its first outbreak still many computers affected by it. Code Red malware are known to exploit Microsoft IIS vulnerability. Code Red infected 265,000 client machines within one day. Total security lost of eliminating code red exceeded 2 billion USD [18].
- Patch management process may degrade the performance of a computer.

Robust Programming Language and Practices

Much vulnerability in the software can be avoided by using robust programming languages and secure software architecture for software development. There are several tools available such as static analysis tools, runtime checking tools for ensuring secure computing environment. Vulnerabilities such as buffer overflow are possible in C, C++ but with use programming language such as Java, Cyclone these vulnerabilities can be avoided. However it is not always possible to work with alternative programming languages because C, C++ provide many low level control which Java, Cyclone can't.

Threat mitigation

As noted in previous section that all vulnerabilities can't be mitigated in time. Therefore threat mitigation is prime necessity to detect and stop malware from spreading in networks. These are several ways organizations can mitigate threat; among them most effective and efficient are Antimalware (including antivirus, anti-spyware), Intrusion Prevention Systems (IPS), firewalls, and router.

Anti-malware

Anti-malware is utility software that is used to detect malware. Anti-malware protection is based on signature and heuristic matching vendors. Signature and heuristics are updated regularly by the anti-malware vendors. These are also new releases to accommodate any new signature generated in the process. Anti Malwares are mostly effective in detecting known worms/virus. There are many vendors providing centralized services of anti-malware. IT-administrator in an organization can install, update such malware in client machines through this centralized console or from their own server. Periodic update can also be managed centrally. Anti-malware software in general has following functionalities. Antimalware Software are mostly effective in detecting known instance of malware, by looking at some fixed sequence of bytes. These sequence of bytes is known as signature. Anti-malware vendors periodically update their signature to detect variations of known worms. However signature fails to detect previously unknown malware. To address new threat anti-malware use heuristic techniques. Heuristic technique look at certain unwarranted behavior or pattern by monitoring system activities, configuration change, network communication and user interaction. However heuristic technique sometimes generate unacceptable false positive i.e. classifying benign program as malicious. False positive may be very much inconvenient to the user. To encounter this problem heuristic capabilities are kept at moderate or low. But this may lead to generate false negative; that is malicious programs are classified as benign. Although antimalware are evolved with

increasing capability they can't display high accuracy in detecting previously unknown threat. That is the reason researchers focus on improving signature accuracy as well as focusing on develop fast automated signature generation for zero-day polymorphic worms.

Network Intrusion Prevention System (NIPS)

NIPS implemented frequently in boarders gateway of networks to perform packet sniffing and network traffic analysis based on certain pre determined rules. NIPS filter the suspicious traffic and allow the acceptable traffic into the network. NIPS uses combination of signature and behavior based technology to filter the exploits. NIPS are well trained about accepted behavior of most vulnerable applications at the gateway such as email servers, web servers etc., therefore can differentiate benign and malicious behavior. NIPS can detect mostly known malware. However NIPS can detect some unknown exploits using application protocol analysis. An example of NIPS is snort-inline (Snort, 2005). NIPS may be customized for the administrator, allowing them to create and deploy signature of new worm. Research focus on this opportunity to build signature of unknown threats and enrich NIPS to handle those exploits.

Firewalls and Routers

Network Firewalls and Routers are the software or devices that decide the entry of network traffic into the system based on certain rules and policies. An example of such rule may be 'drop all TCP connection that are sourced from a particular IP addresses. Firewalls are of two types: Network firewalls and host based firewalls. Network firewalls deployed at the gateway of a network to decide which traffic to pass from one network to the other network. Network firewalls can monitor both incoming and out coming traffic, thereby preventing external threats to enter into the system and internal exploits to spread the external network. This is achieved by performing Ingress and Egress filtering. Ingress filtering blocks incoming malicious traffic to enter into the system and Egress filtering prevents outgoing unwarranted traffic to exit the network. Network firewalls frequently engaged in Network Address Translation (NAT). NAT is mechanism of mapping addresses of one network onto addresses in other network. NAT map private addresses of internal network to public addresses of connected networks. If an external host is compromised it can't initiate direct connection to internal host as private addresses are not routable across the internet. This characteristic also limits the discovery of valuable host outside private address space through scanning. Host based firewalls are installed on individual host to monitor incoming and outcoming traffics.

Worm Detection

Based on technique and parameter used worm detection can be broadly classified as anomaly based detection. In anomaly based detection the detector use the definition of normal behavior of network. Any deviation is treated as anomaly. Signature based detection on the other hand is based on certain pattern in worm-payload.

Anomaly based detection

Anomaly based detection (Kruegel and Vigna, 2003; Wang and Stolfo, 2004) is performed in two steps. In first step the detector is trained to understand what normal or acceptable network behavior is. Second step is detection, based on any deviation from acceptable behavior. One special type of anomaly based detection is specification based detection, which define normal network behavior based on some specifications or rules. The study of existing worm is an important training phase. It has been observed that fast spreading worms creates large traffic. Some worms target nonexistent IP address and closed ports. While normal traffic behaves consistently, malicious traffic may disturb this consistency. Anomaly based detection does not look at the payload content of worm instead monitor network traffic volume, type of connection or host behavior. Apart from detecting known attack anomaly based detection is also useful in detecting some zero- day unknown attacks whose signature is yet to be created. Major limitations of anomaly based detection are that it is difficult to cover all specification or rules of normal behavior therefore generating huge false alarm. Both the false positive and false negative rate may be high. As the behavior is a qualitative feature it is difficult to set threshold when alarm will be generated. Anomaly-based detection may be classified based on connection attempts, where anomaly is triggered based on connection count, failure connection rate, success/failure connection rate. Illegal traffic may be diverted and monitored at darknets or honeypots. Anomaly may also be found in packet payload, by comparing with model, learned during training phase (Li *et al.,* 2008).

Signature Based Detection

Signature based detection (Kim and Karp, 2004; Kreibich and Crowcroft, 2003; Newsome *et al*., 2005; Newsome and Song, 2005) look for traffic payload for particular pattern or sequence of byte that is unique in a worm. This unique pattern or sequence of byte defines the signature of worm. Although there are different definitions of signature most common are content based signature. Signature based detection is very popular intrusion detection mechanism. Many commercial anti-malware systems employ this technique along with some heuristic technique. Signature based approach does not require the knowledge of worm target discovery method, propagation scheme instead it is focused on particular content in its payload. Every packet is matched with signature stored in database; alarm is generated when match found. Signature based approaches are mostly effective in detecting known however recent research strongly focus on generating automated signature for zero day worm including polymorphic and metamorphic worms (Newsome *et al.,* 2005; Paul and Mishra, 2013). The current limitations of signature based approaches are

- With increase in worm occurrence, the number of signature is also growing. There is a chance of performance bottleneck for the signature processing engine.

- Signature based approach are most effective in detecting known attack. However there are considerable research efforts has been given to generate for new attacks too.
- Most of the signature based approach for known worm based on either string or regular expression matching. This method can be deceived by the attacker.
- Signature generation is still not fully automatic. These are many cases where signature is generated with manual effort. The process is time consuming and largely depends on experts experience and judgment.

A part of our research will address the above mentioned problem of signature generation.

Analysis

Organization should have mechanism to detect the malware incidents at early stage to prevent spreading it like epidemic. Cost of damage and recovery effort may also be minimized substantially. Once the malware is detected it is very important to analyze the code to find out any unique characteristics which define that class of malware.

Worm Containment

Preventing malware incidents and detecting them are two important steps of defense mechanism. But it is also important to contain the malware to stop them at early stages to prevent infection at epidemic rate. Once the worm is detected, the system usually tunes the following parameters to slow down the aggression: reaction speed, containment strategy and deployment scenario. Following are the broad categories of worm containment methods.

- Slowing down the speed of worm propagation is one strategy to give the security personel enough time for intervention. Dantu and Yelimeli (Dantu and Yelimeli, 2004)proposed the dynamic control of worm propagation to delay the malicious traffic by using feedback loop. Wong *et al* proposed dynamic quarantine of worms (Wong *et al.*, 2004) to slow down worm spread. It has been observed that worm propagation speed is much higher than human reaction, therefore slowing down may not stop them from spreading.
- Blocking selective connections or traffic is another approach of containment. Whenever any anomalous behavior is detected the source has to be blocked to stop further infection attempt.GU *et al* (Gu *et al.,* 2004) taken the approach of blocking connection from host discovered to be infected. Blocking connection may generate false alarm therefore to be managed carefully. Content blocking is another effective way of stopping the spread of worms. If the signature based IDS find match in a packet, then the packet is automatically dropped allowing the legitimate traffic to pass through.
- Generating patch for a recently discovered vulnerabilities and distributing them to the machines that have this vulnerability is another way of mitigating further damage. Sigiroglou *et al* (Sidiroglou and Keromytis, 2005) uses sand-box technique to generate patch for vulnerable machines to prevent the infection spreading the production system and leave the identified worm to a third party system such as honeynet, IDS etc.
- Honeypot based systems place a vulnerable host on the network that provides no real services. Honeypot lure the worms. Any traffic trapped into honeypot may be considered suspicious. There are several open source and commercial honeypot available. HoneyD (Provos, 2004; Kreibich and Crowcroft, 2003) is one such open-source honeypot widely used in research. HoneyD is able to simulate large network address space luring the worms into it. This strategy slows down the infection into production network. Moreover security experts also get sufficient time to analyze the trapped malware thus generating signature may be easy and more accurate. Burmley *et al.* pointed that local containment will only be effective if the deployment ration is high. (If we can deploy local containment in 50% of hosts in internet, propagation may be slowed down by a factor of 2) (Brumley *et al.*, 2005). They have also concluded that hybrid approach of proactive protection and signature based detection may provide desired level of security from most sophisticated worms.

Conclusion

We have presented in this paper, the generic definition of malware and their general characteristics. Malware lifecycle is being described in terms of target discovery, malware propagation, and payload activation. Different mechanisms employed in this category are discussed in this paper. Author can focus on these features of malware to develop more robust defense system. We have also discussed the worm defense life cycle in terms of prevention, detection, analysis and containment. Each element of defense has their merits and limitations. Researcher can focus on the challenges of these mechanisms. Many a times a hybrid approach may be more effective in protecting vulnerable cyber space.

REFERENCES

Bolzoni, S. E. D. and Hartel, P. 2006. "POSEIDON: A 2-Tier Anomaly- Based Network Intrusion Detection System," *Proc. 4th IEEE Int'l Wksp. Info. Assurance*,

Brumley D. *et al.*, "Design space and analysis of worm defense strategies" , in Proc. ACM ASIACCS, 2006, PP. 125-137.

Dantu, J. C. R. and Yelimeli, A. 2004."Dynamic Control of Worm Propagation," *Proc. Int'l Conf. Info. Technology: Coding and Comp.*,

eEye Digital Security. .ida "Code Red" Worm, http:// www.eeye.com/html/research/advisories/ al20010717. html.

F-Secure. "F-Secure Corporation Virus Glossary". http://www.f-secure.com/virus-info/glossary.shtml, May 2003.

Gamespy. Gamespy arcade, http://www.gamespyarcade.com.

Gu, G., Sharif, M., Qin, X., Dagon, D., Lee, W. and Riley, G. 2004. \Worm detection, early warning and response based on local victim information," in Proceedings of the Annual Computer Security Applications Conference. Washington, DC: IEEE Computer Society,

Kim H.A. and B. Karp. Autograph: toward automated, distributed worm signature detection. In Proceedings of the 13th USENIX Security Symposium, August 2004.

Jason V Miller, Jesse Gough, Bartek Kostanecki, Josh Talbot, and Jensenne Roculan. Microsoft dcom rpc worm alert, https://tms.symantec.com/members/ analystreports/030811-alert-dcomworm.pdf.

Jiankun Hu and Xinghuo Yu, 2009. "A Simple and Efficient Hidden Markov Model Scheme for Host-Based Anomaly Intrusion Detection" *IEEE Network Journal,* Volume 23 Issue 1, January/February

Kreibich, C. and Crowcroft. J. 2003. Honeycomb – creating intrusion detection signatures using honeypots. In Proceedings of the Second Workshop on Hot Topics in Networks (HotNets-II), November.

Kruegel, C. and Vigna, G. 2003 "Anomaly Detection of Web-based Attacks'' in Proc. ACM Conference Computer and Communication Security, ACM , pp. 251-261.

Li, P. Salour, M. and Su, X. 2008. "A survey of internet worm detection and containment", Communications Surveys and Tutorials, IEEE, vol. 10, pp. 20-35,

Mark Eichin and Jon Rochlis.1989. With Microscope and Tweezers: An Analysis of the Internet Virus of November 1988. In IEEE Computer Society Symposium on Security and Privacy,

Markus Kern. Re: Codegreen beta release, http:// online.securityfocus.com/archive/82/211462.

Mishra, B.K. and Saini, D. K. 2007. "SEIRS epidemic model with delay for transmission of malicious objects in computer network," Applied Mathematics and Computation, Elsevier, 188, pp.1476–1482.

Mohsen Damshenas *et al.,* 2013. A Survey on Malware propagation, Analysis and Detection, International Journal of Cyber Security and Digital Forensic(IJCSDF), 2(4), pp. 10-29,

Moore, V. P. D. *et al.*, 2003. "Inside the Slammer Worm," *IEEE Sec. and Privacy*, Vol. 1, pp. 33–39.

Provos N. 2004. "A Virtual Honeypot Framework," *Proc. 13th USENIX Sec. Symp.*,

Network Associates, "Virus Glossary". http://mcafeeb2b.com/ naicommon/avert/avert-researchcenter/virus-glossary.asp, 2003.

Newsome, J. and Song, D. 2005. Dynamic taint analysis for automatic detection, analysis, and signature generation of exploits on commodity software. In Proceedings of the 12th Annual Network and Distributed Systems Security Symposium, February.

Newsome, J. Karp, B. and Song, D. 2005. Polygraph: Automatically generating signatures for polymorphic worms. In Proceedings of the IEEE Symposium on Security and Privacy, May

Paul, S.and Mishra, B.K. 2013. "PolyS-Network based signature generation for zero-day polymorphic worms" SERSC International Journal of Grid and distributed computing, Vol 6, No. 4, pp.63-74, 2013.

Rhoades, K. 2001. "Code Red, Code Red II, and SirCam Attacks Highlight Need for Proactive Measures", August 21, Available: http://www.gao.gov/new.items/d011073t. pdf.

Sidiroglou S. and Keromytis, A D. 2005. Countering network worms through automatic patch `generation. *IEEE Security and Privacy,* 3(6): 41 49, November

Stuart Staniford, Vern Paxson, and Nicholas Weaver, 2002. How to 0wn the Internet in Your Spare Time. In Proceedings of the 11th USENIX Security Symposium. USENIX, August

Wang, K. and Stolfo, S. J. 2004. "Anomalous Payload-based Network Intrusion Detection.'' in proc. 7th International Symposium on Recent Advances in Intrusion Detection (RAID ').

Weaver, V. P. N., Staniford, S. and Cunningham, R. 2003. "A Taxonomy of Computer Worms," *Proc. ACM WORM '03*,

Wong, C. W. C. *et al.* 2004. "Dynamic Quarantine of Internet Worms," *Proc. Int'l Conf. Dependable Sys. and Networks*.

CERT. 1999. CERT Advisory CA-1999-04 Melissa Macro Virus, http://www.cert.org/advisories/ca-1999-04.html.

CERT. 2000. CERT Advisory CA-2000-04 Love Letter Worm, http://www.cert.org/advisories/ca-2000-04.html

Snort, May 2005, http://www.snort.org

F-Secure Virus Descriptions: Nimda, retrieved July 2007, http://www.f-secure.com/v-descs/nimda.shtml, 2001.

****