AN ANALYSIS OF PRE-INFECTION DETECTION TECHNIQUES FOR BOTNETS AND OTHER MALWARE

Mark Graham
mark.graham@anglia.ac.uk
Department of Computing and Technology
Anglia Ruskin University
Cambridge, CB1 1PT

Adrian Winckles
adrian.winckles@anglia.ac.uk
Department of Computing and Technology
Anglia Ruskin University
Cambridge, CB1 1PT

ABSTRACT

Traditional techniques for detecting malware, such as viruses, worms and rootkits, rely on identifying virus-specific signature definitions within network traffic, applications or memory. Because a sample of malware is required to define an attack signature, signature detection has drawbacks when accounting for malware code mutation, has limited use in zero-day protection and is a post-infection technique requiring malware to be present on a device in order to be detected.

A malicious bot is a malware variant that interconnects with other bots to form a botnet. Amongst their multiple malicious uses, botnets are ideal for launching mass Distributed Denial of Services attacks against the ever increasing number of networked devices that are starting to form the Internet of Things and Smart Cities. Regardless of topology; centralised Command & Control or distributed Peer-to-Peer, bots must communicate with their commanding botmaster. This communication traffic can be used to detect malware activity in the cloud before it can evade network perimeter defences and to trace a route back to source to takedown the threat.

This paper identifies the inefficiencies exhibited by signature-based detection when dealing with botnets. Total botnet eradication relies on traffic-based detection methods such as DNS record analysis, against which malware authors have multiple evasion techniques. Signature-based detection displays further inefficiencies when located within virtual environments which form the backbone of data centre infrastructures, providing malware with a new attack vector. This paper highlights a lack of techniques for detecting malicious bot activity within such environments, proposing an architecture based upon flow sampling protocols to detect botnets within virtualised environments.

KEY WORDS: Malware Detection Evasion, Botnet, C&C, P2P, Flow Sampling, Virtual Environment

1.0 INTRODUCTION

Recommended best practise for protecting internetworked devices from malicious software is to deploy host-based real-time anti-malware (anti-virus, anti-spyware, anti-rootkit) detection onto network end points. Historically this technique has proven to be successful in detecting malware and disinfecting compromised devices. Today it is less successful when trying to protect against rapidly mutating strains of malware variants. In May 2014, Symantec (2014) suggested that anti-virus software now has such a poor detection rate, capturing only 45% of modern attacks, that this technique should now be considered "dead."

One of the most serious malware threats faced by organisations today comes from malicious bots. Malicious bots work together to create a botnet; a network of infected host devices which acts as a single platform from which to launch a massive co-ordinated attack. Botnets consist of many thousands of infected hosts which receive instructions from Command and Control (C&C) servers operated by a human botmaster. Malware authors utilise many traditional virus concealment techniques to allow their bots to evade detection systems.

Whilst host-based anti-malware detection systems are used to protect end points from bots, this type of system has little impact when it comes to eradicating the overall botnet. Removing a single compromised device from the botnet of thousands of infected devices is negligible when a botnet can recruit ten replacements. Eradication of a botnet can only be achieved through takedown of the C&C server. Bots differ from traditional malware in that, once resident on a victim, they periodically communicate with their C&C server for updates and attack instructions. Analysis of these communication packets can be used to create a topology map of the overall botnet. Interpretation of the flow of communication within this topology map should identify the C&C server, aiding in the takedown of the botnet.

The past few years have seen an evolution in the way IT departments provide services. For reasons of cost and ease of management many services and systems that were once hosted within an organisation's datacentre have been relocated to the cloud. It is common place for the infrastructure for these types of cloud services to be built upon virtualised environments so as to better utilise expensive hardware. Such cloud services are the building blocks for the Internet of Things and Smart City infrastructures.

The contributions of this paper are threefold:

1) This paper highlights the drawbacks of traditional signature-based anti-virus detection in protecting against bot-style malware.
2) The alternative to signature-based detection is signature-less detection which is based upon traffic analysis. Today many service providers rely on DNS record analysis to identify botnet activity. This paper discusses the techniques available to malware authors to evade DNS based detection and considers flow protocols, such as NetFlow, as an alternative detection tool.
3) Finally, this paper raises the lack of research being done into botnet detection within virtual environments, the building blocks of the next generation of the Internet. We propose an architecture for botnet detection within virtual environments based upon flow.

We start this paper by introducing the shortcomings in signature-based anti-virus detection techniques in section 2. Section 3 presents traffic-based signature-less detection. In section 4 we consider flow as a method to detect botnets within virtual environments and we conclude the paper in section 5.

2.0 SIGNATURE-BASED DETECTION

Anti-virus protection forms an integral part of computer and network security to detect and subsequently remove malicious programs. Detection software is situated on either a) PCs or servers to protect the end-points themselves, or b) on edge devices positioned between the host networks and the cloud so as to protect multiple hosts on a network and act as an early detection system. Edge devices such as Intrusion Prevention Systems, gateways, firewalls or email servers are all equally vulnerable to evasion and are unable to detect malware that enters the network through other attack vectors such as mobile internet connections, or removable devices.

The simplest way to detect malware is signature-based detection. Once compiled, malware source code produces a binary executable. The binary is a hexadecimal file, unique to that source code, like a fingerprint or signature that characterises the malware. Signature-based detection compares suspect code against known malicious code signatures. Even a slight change to the malware source code requires the malware researcher to create a new signature even if the malware attack profile remains identical, thus requiring anti-virus software to be constantly updated with the latest definition files. An estimated 30 million new malware strains were in circulation in 2013, at an average of 82,000 per day, bringing the total number of known malware samples to 145 million (PandaLabs, 2013).

Signature-based detection has three significant drawbacks:

a) **An inability to cope with malware variants** – each tweak to a virus binary requires a new signature definition to be created
b) **A lack of zero day protection** – an attack window exists between a new virus strain being released into the wild and end-point anti-virus software obtaining said malware's corresponding signature definition
c) **A post-infection technique**
   i) a sample of the malware variant is required in order to create the signature definition
   ii) the malware must attempt to infect the device in order for the detection system to have the opportunity to recognise the malicious signature

2.1 Binary Obfuscation

To evade signature-based detection, malware authors employ techniques to change the signature pattern of an existing virus, thus creating a new variant or strain.

2.1.1 Encryption

Malware code can be obfuscated from detection systems by utilising XOR-based symmetric encryption/decryption ciphers (You and Kim, 2010). Encryption has two effects; not only does it make the malware more difficult to detect it also makes the malware considerably more difficult to reverse engineer, a vital step in creating a signature definition.

Malware must decrypt its binary in order to execute. To do this, malware must carry a decryptor engine. Malware detection is usually through indirect recognition of the code pattern of the decryptor engine, rather than the obfuscated malware source code.

2.1.2 Code Mutation

Mutation techniques make minor adjustments to the malware binary whilst keeping the original attack vector algorithm intact. Although minor, these adjustments are usually significant enough to warrant the creation of a brand new signature definition for that variant, thereby creating a zero-day malware which is not yet detectable by signature-based anti-virus systems.

Malware mutation falls into two categories:

a) Encryption/decryption engine algorithm mutation:
   Oligomorphism - Malware mutates its engine by randomly choosing from pre-defined engines
   Polymorphism - Malware creates a new, unique engine

b) Code structure mutation:
   Metamorphism - Malware code structure is changed using techniques such as registry reassignment, noise insertion, null code insertion or redundant code insertion. Self-propagating metamorphic malware will most likely carry its own morphing engine to remain undetectable.
   Evolution Algorithms – Cani et al. (2013) mimicked natural biological virus evolution, using stochastic search engine optimisation algorithms to create malware capable of evading heuristic detection. No real world malware is known to use these techniques.

Signature-based detection is more likely to detect the morphing engine than the mutated malware (Sridhara and Stamp, 2013). Hidden Markov Models (HMM) are able to detect metamorphic malware using statistical pattern analysis. Lin and Stamp (2011) were able to exploit weaknesses in HMM-based detection to create highly metamorphic viruses, but with 70% of the original source code requiring replacement, this practically of this method is questionable.

2.2 Heuristic Detection

Heuristic detection utilises signature-based detection principles, but rather than scanning for specific strains of malware, heuristics look for similar, more general behavioural characteristics across a malware family. Machine learning is used to identify patterns. This less exact matching method has the advantage of being able to detect zero-day malware. A major downside of heuristic detection is the high number of false positives detected. Additionally, heuristic scanning and analysis can be a lengthy process which can impact the performance of the device running the heuristic detection system.

2.3 File Parsing Exploitation

Modern anti-virus scanners are required to parse files in a multitude of different formats. Discrepancies in the way file-parsing semantics differ between applications and operating systems (OS) are exploitable by amending a file's metadata to turn the file into a format unrecognisable by the scanner, forcing anti-virus scanners to misunderstand the structure of suspicious files. These vulnerabilities were unrelated to malware obfuscation, mutation, binary packing, or other code manipulation stealth techniques. Parsing files of certain formats is notoriously difficult, such as HTML which can contain many irrelevant characters. Signature-based anti-virus detection is known to ineffectively process malformed archive file formats which require extraction before they can be scanned. Unmodified, easily detectable viruses can evade detection using simple file-parsing exploits.

Anti-virus scanners are vulnerable to file parsing exploitations (Jana and Shmatikov, 2012):

a) ***Chameleon attack*** – exploits heuristic parsing discrepancies where files appear as one type to the detector and as a different type to the operating system
b) ***Werewolf attack*** – exploits discrepancies in parsing of executables and application specific formats, because the attack files appear to have a different structure depending on if they are parsed by the detector or the application

3.0 TRAFFIC-BASED DETECTION

Although the attack vectors of botnets vary, all botnets exhibit similar behavioural characteristics which involve information exchange with the C&C server(s). Holistic observation of this communication traffic may provide evidence to render the malware detectable. Common across all botnets are three information exchanges with the C&C server:

a) ***Recruitment*** - in order to grow, botnets must recruit new members to replace those that have been disinfected
b) ***Maintenance*** – such as updates and upgrades, to ensure the botnet remains undetectable from anti-virus detection
c) ***Attack*** – the raison d'etre of a botnet is a mass attack on a victim. Botnets are designed to be stealthy. It is during the attack phase that they are most detectable. The attack requires two information exchanges: the attack commands to initiate the attack and the attack itself.

A number of different approaches to detecting malicious activity via traffic monitoring techniques have been studied. These all tend to be centred on detecting anomalous behaviour, which involves capturing a network traffic component and comparing it to either a baseline of normal behaviour or a formal definition of what normal and abnormal behaviours are. Because this does not require creation of a malware specific signature, it is not prone to the issues associated with signature-based detection outlined in section 2. The disadvantage of such techniques, however, is defining "normal".

3.1 DNS Record Analysis

To be able to communicate with their C&C server, bots need to know the server's IP address. Early bots carried this IP address within their binary, which meant the C&C server could be traced through binary reverse engineering. DNS record analysis takes advantage of the need for modern bots to resolve the C&C server domain name to an IP address. Botnets are usually detected by the very techniques they utilise in an attempt to evade DNS detection. Malware authors use fluxing techniques to disassociate DNS records with the C&C server IP address. Frequent DNS record changes are a strong indicator of potential bots. DNS records with short Time-To-Live (TTL) indicate volatile DNS records, which is further evidence of botnet-type activity. DNS analysis techniques can lead to false positives. Many legitimate domains also have low TTL values in the DNS records. Villamarin-Salomon and Brustoloni (2008) suggest analysis of recurring NXDOMAIN replies to DNS queries as a more effective method for botnet detection compared to TTL analysis.

DNS detection evasion, or fluxing, techniques include:

a) ***IP Fluxing*** - rapidly registers and deregisters thousands of IP addresses associated with a single fully-qualified domain names
b) ***Domain Fluxing,*** or fast-fluxing - the inverse of IP fluxing, which changes the domain name of the C&C server quicker than it can be traced. Dynamic Generation Algorithms [DGAs] are used to create a wide range of pseudo-random domain names. Blacklisting DGA domain names is impossible because i) these algorithmically generated domain names have a high entropy of randomly chosen alphanumeric characters which are very different to human generated domain names, ii) these domains are short lived and iii) only a few of these domain names actually host a C&C server so as mitigating against DGA being deciphered if the malware binary is reverse engineered. Conficker, Torpig and Kraken are bots that used DNS fast-fluxing.

Fast-flux botnet detection has been studied in depth. Fast-flux botnets can be detected using the number of distinct Address records and the number of different Autonomous System Numbers associated with the domains (Perdisci et al., 2009). Stalmans and Irwin (2011) furthered fast-flux DGA botnet detection by using Bayesian analysis to determine if a domain is legitimate or malicious. However, this still produces 10 percent false positives. Yadav and Reddy (2012) used failed domain names to detect both C&C servers and bots within a network.

3.2 Flow Traffic Analysis

A C&C botnet has a centralised client-server topology, comprising of a few servers and a multitude of bot clients. Centralised C&C topology was common in early IRC and HTTP bots. Because C&C bots must resolve the IP address of their C&C server in order to communicate with it, DNS record analysis can be used in C&C botnet detection. Many modern botnets utilise a decentralised Peer-to-Peer (P2P) topology. Storm, Nugache, and Conficker showed embedded P2P capabilities. In a P2P topology each bot is capable of acting as a C&C server, greatly increasing the botnet’s resilience against takedown. DNS record analysis is less effective for P2P botnet detection due to the vast number of bots that could possibly act as a server. P2P bots still need to interact. In doing so, they still generate sufficient communications traffic which can be used to detect the botnet working on the assumption that P2P bots within the same botnet share similar C&C communication patterns (Gu et al., 2008).

A potential alternative to DNS record analysis for detecting P2P botnets is flow traffic analysis. NetFlow (Kerr and Bruins, 2001) is a network traffic monitoring protocol developed to provide enhancements over Simple Network Monitoring Protocol (SNMP). Karasaradis, Rexroad and Hoeflin (2007) first proposed using NetFlow in bot detection, successfully detecting IRC bots but not P2P or HTTP bots. Notable studies into flow-based detection include BotCloud (Francois et al., 2011) which proposed a modified PageRank algorithm to make sense of the enormous quantities of data that NetFlow can produce. Aggregation of NetFlow fields such as source & destination IP address, source and destination port, packet event times and bytes per packets can be used to characterise bot activity in a simulated Local Area Network (Amini, Azmi and Araghizadeh, 2014). The full potential for sampling flow protocols such as NetFlow and IPFIX versus mirroring flow protocols such as sFlow in botnet detection is not yet fully understood.

4.0 BOTNET DETECTION IN VIRTUAL ENVIRONMENTS

Cloud computing is widely accepted as a service for providing offsite computer processing and storage. The infrastructure for such services, including software-as-a-service (SaaS), platform-as-a-service (PaaS) and infrastructure-as-a-service (IaaS), typically employ virtualisation techniques to recognise increased hardware utilisation efficiencies in CPUs, memory, storage and input/output. These platforms will form the basis of the Internet of Things and future Smart Cities. Little research into understanding botnet detection techniques within virtual infrastructures can be found.

Signature-based detection displays an additional drawback when considered from the perspective of virtualisation. Signature-based detection takes place at the application level within the OS. Rootkit malware operates at the OS kernel level, altering information provided by the OS to the detection software thereby masking signs of the malware’s presence or turning off detection systems altogether (Watson, 2012). Malware detection systems deployed within-OS will always be vulnerable to such exploitation. A virtual environment offers the opportunity to relocate a detection system outside-of the OS where is it not vulnerable to this exploit. A system for detecting botnets within virtual environments should not be based upon signature-based detection systems for the reasons outlined in section 2. The alternative is a traffic-based detection system, which does not rely on the deep packet inspection techniques which signature-based detection utilises in order to obtain the code to be scanned. Such traffic-based detection is not subject to data privacy concerns and therefore can be deployed in cloud provider data centres. Existing traffic-based detection systems such as DNS record analysis will continue to be vulnerable to fluxing exploitations outlined in section 3.1 and ineffective for P2P botnet detection as outlined in section 3.2.

Watson (2012) theorised that a detection system for worm-type malware in virtual environments may be possible using VMM memory introspection combined with a heuristic or statistical detection algorithm to compare a virtual machine (VM) snapshot against an earlier malware-free snapshot. Whilst this technique may be adaptable to detect botnet-style activity for VM disinfection it lacks the features of traffic based detection systems required for C&C takedown.

We propose a system for detecting botnets within virtual environments based upon flow protocols such as NetFlow, IPFIX or sFlow. Section 3.2 highlights several studies that have been successful in detection botnets using NetFlow. If botnets exhibit similar behaviour in virtual environments as they do in physical environments then the success of NetFlow as a detection tool can be extrapolated into virtual infrastructures. Flow is a sampling protocol, therefore flow collection points can be situated anywhere in a virtual environment. A virtual infrastructure offers a number of potential flow sampling points from which to collect flow traffic parameters.

Figure 1 identifies four potential sample collection points:

1. on a ***virtual interface*** - to collect potential malicious traffic exiting or entering a VM
2. on a ***virtual switch*** - to collect potential malicious traffic communications between VMs
3. on a ***virtual router*** - to collect potential malicious traffic communications between virtual environments
4. on a ***hypervisor*** - to collect potential malicious traffic communication between network bridges connecting VMs

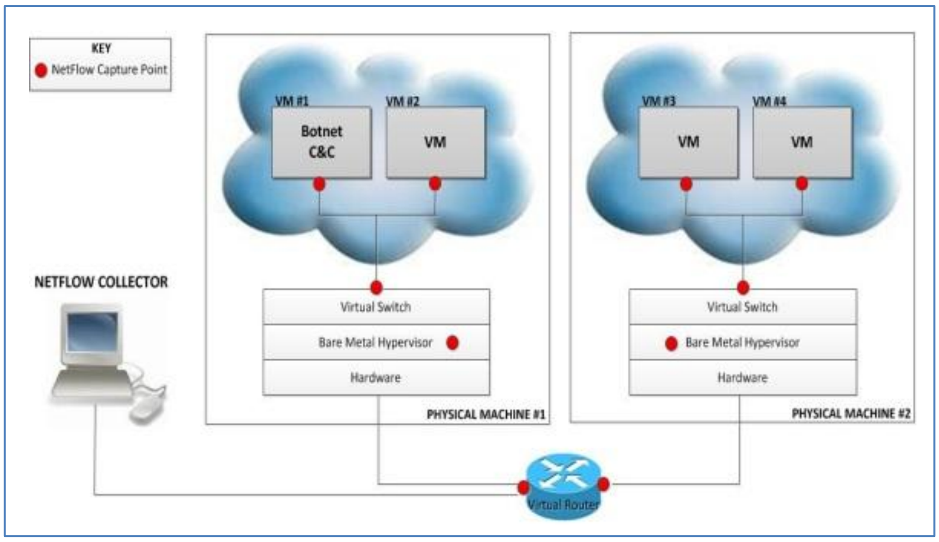

Figure 1: Architecture for Botnet Detection in Virtual Networks

5.0 CONCLUSION

Signature-based detection is an extremely reliable technique when detecting *known* viruses, worms and Trojans. Disinfecting a single compromised device has little impact towards botnet eradication. Defence against botnets necessitates the detection and takedown of C&C server(s), which is better suited to signature-less traffic-based detection such as DNS record analysis. DNS record analysis is successfully used today by anti-malware organisations across the globe, but this paper demonstrates that botnet authors have techniques to evade and delay detection through DNS.

Virtualisation is now all pervasive across the field of information technology. As the world adopts more and more IP enabled devices as the Internet of Things moves ahead, little research is being done towards botnet detection in virtual environments. Watson (2012) theorised that VMM memory introspection with a heuristic or statistical detection technique could detect worm-type malware in virtual environments. This technique may detect botnet-style activity, but would contribute little towards C&C takedown.

Having recognised a lack of knowledge, this paper starts to address the theoretical considerations for botnet detection within a virtual environment. In section 4 this paper proposes that flow protocols could be used to create a traffic-based detection model that is not subject to the issues outlined within. An initial proof of concept network has successfully recognised traffic signatures for Zeus bot captured via NetFlow between two virtual machines. Future work will focus on the development of a distributed packet capture monitoring function based on open source tools in order to detect C&C and P2P malware evident in a virtual environment.

In conclusion, flow traffic analysis is one technique that may prove useful in the fight for botnet detection in virtual environments and warrants further investigation.

REFERENCES

Amini, P., Azmi, R. and Araghizadeh, M., 2014. Botnet Detection using NetFlow and Clustering. *Advances in Computer Science: an International Journal*, 3 (2), pp.139-149.

Cani, A., Gaudesi, M., Sanchez, E., Squillero, G. and Tonda, A., 2013. Towards Automated Malware Creation: Code Generation and Code Integration.

Francois, J., Wang, S., Bronzi, W., State, R., and Engel, T., 2011. BotCloud: detecting botnets using MapReduce. *International Workshop on Information Forensics and Security (WIFS) 2011.* IEEE, pp. 1 -6.

Jana, S. and Shmatikov, V., 2012. Abusing file processing in malware detectors for fun and profit. *IEEE Symposium on Security and Privacy 2012.* IEEE, pp. 80-94.

Karasaridis, A., Rexroad, B. and Hoeflin, D., 2007. Wide-scale botnet detection and characterization. *Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, Vol. 7, 2007.*

Kerr, D., Bruins, B., Cisco Systems, Inc., 2001. *Network Flow switching and flow data export*. U.S. Pat. US6243667 B1.

Lin, D. and Stamp, M., 2011. Hunting for undetectable metamorphic viruses. *Journal in Computer Virology*, 7 (3). Springer, pp.201-214.

Panda Labs, 2013. Annual Report PandaLabs 2013 Summary, [online] Available at: <http://press.pandasecurity.com/wp-content/uploads/2010/05/PandaLabs-Annual-Report_2013.pdf> [Accessed 6 May 2014].

Perdisci, R., Corona, I., Dagon, D. and Lee, W., 2009. Detecting malicious flux service networks through passive analysis of recursive dns traces. *Computer Security Applications Conference, 2009. ACSAC'09. Annual.* IEEE, pp. 311-320.

Sridhara, S.M. and Stamp, M., 2013. Metamorphic worm that carries its own morphing engine. *Journal of Computer Virology and Hacking Techniques*, 9 (2), pp.49-58.

Stalmans, E. and Irwin, B., 2011. A framework for DNS based detection and mitigation of malware infections on a network. *Information Security South Africa (ISSA), 2011.* IEEE, pp. 1-8.

Symantec, 2014. Symantec: Antivirus is 'DEAD' – no longer 'a moneymaker', [online] Available at: <http://www.theregister.co.uk/2014/05/06/symantec_antivirus_is_dead_and_not_a_moneymaker/> [Accessed 7 May 2014].

Villamarín-Salomón, R. and Brustoloni, J.C., 2008. Identifying botnets using anomaly detection techniques applied to DNS traffic. *Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE*. IEEE, pp. 476-481.

Watson, M. R., 2012. Malware Detection in the Context of Cloud Computing. *The 13th Annual Postgraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting.*

Yadav, S. and Reddy, A.N., 2012. Winning with DNS failures: Strategies for faster botnet detection. *Security and Privacy in Communication Networks.* Springer, pp.446-459.

You, I. and Yim, K., 2010. Malware Obfuscation Techniques: A Brief Survey. *Broadband, Wireless Computing, Communication and Applications (BWCCA), 2010 International Conference*. IEEE, pp. 297-300.