ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Privacy Ensured Domain Name Query Scheme with Attack Detection Mechanism

M. Dhaarani[1], Mr. S. Sivaraj, ME, [2]

II-M.E (CSE), Dept. of CSE, SSM College of Engineering, Komarapalayam, Tamilnadu, India[1]

Assistant Professor, Dept. of CSE, SSM College of Engineering, Komarapalayam, Tamilnadu, India[2]

Abstract: Distributed Denial of Service (DDoS) attacks are initiated by the botnets. Botnets are managed by the central authority. Botnets are capable to initiate many DoS attacks. Scanning, DoS Attacks, Sniffers, Information harvesting, Encryption are the main operations of the botnet. AgoBot, SDBot, SpyBot, and GTBot are the most commonly used bot families. Denial-of-Service (DoS) attack is an attempt by attacker to prevent legitimate users from using resources. Denial-of-Service denies a victim (host, router, or entire network) from providing or receiving normal services. Distributed Denial of Service (DDoS) Attacks are generated in a "many to one" dimension. In DDoS attack model large number of compromised host are gathered to send useless service requests, packets at the same time. Botmaster instructions are distributed through hidden channels in a network. A C&C channel for a botnet needs to be reliable, redundant, noncentralized and easily disguised as legitimate traffic. Domain Name Service (DNS) provides a distributed infrastructure for storing, updating and disseminating data. DNS is targeted as a stealthy botnet command-and-control channel. Malicious DNS activities are hide at the network level. Exponentially Distributed Query and Piggybacking Query attacks are detected using the markov chain analysis and statistical analysis mechanism. Probability distribution based analysis model is used to detect automatic domain flux attacks. DNS tunneling technique is used for transmitting arbitrary data via DNS protocol. Network flow based attacks are controlled with secured data communication through DNS between the nodes. Automated anomaly detection is adapted to the system. Navy bayesian classification technique is integrated to the system. Small query analysis mechanism is integrated with the system.

I. INTRODUCTION

Intrusion detection systems are the 'burglar alarms' of the computer security field. The aim is to defend a system by using a combination of an alarm that sounds whenever the site's security has been compromised, and an entity—most often a site security officer (SSO)—that can respond to the alarm and take the appropriate action, for instance by ousting the intruder, calling on the proper external authorities, and so on. This method should be contrasted with those that aim to strengthen the perimeter surrounding the computer system. We believe that both of these methods should be used, along with others, to increase the chances of mounting a successful defence, relying on the age-old principle of defence in depth. In anomaly detection we watch not for known intrusion—the signal—but rather for abnormalities in the traffic in question; we take the attitude that something that is abnormal is probably suspicious. The construction of such a detector starts by forming an opinion on what constitutes *normal* for the observed subject, and then deciding on what percentage of the activity to flag as *abnormal*, and how to make this particular decision. This detection principle thus flags behaviour that is unlikely to originate from the normal process, without regard to actual intrusion scenarios.

Denial-of-service attacks are considered violations of the Internet Architecture Board's Internet proper use policy, and also violate the acceptable use policies of virtually all Internet service providers. They also commonly constitute violations of the laws of individual nations. Denial-of-service attacks can also lead to problems in the network 'branches' around the actual computer being attacked. For example, the bandwidth of a router between the Internet and a LAN may be consumed by an attack, compromising not only the intended computer, but also the entire network or other computers on the LAN. If the attack is conducted on a sufficiently large scale, entire geographical regions of Internet connectivity can be compromised without the attacker's knowledge or intent by incorrectly configured or flimsy network infrastructure equipment.

A DDOS attack (a Distributed Denial of Service attack) is a type of web attack that seeks to disrupt the normal function of the targeted computer network. This is any type of attack that attempts to make this computer resource unavailable to its users. While this type of attack typically follows the same sorts of patterns, the definition of the term Distributed Denial of Service does not make any specific indications of how this type of attack is to be pulled off. What makes this type of attack "distributed" is the concerted efforts between a large number of disruptors all for the common goal of preventing web servers from functioning effectively at all. These users may be willing participants, or in some cases be tricked into downloading software that will use their terminal to aid in the offensive. All in all, regardless of the means, a DDOS attack is simply a combined effort to prevent computer systems from working as well as they should, typically from a remote location over the internet.

The most common method of attack is to send a mass saturation of incessant requests for external communication to the target. These systems are flooded with requests for information from non-users, and often non-visitors to the website. The goal of this attack is to create a large enough presence of false traffic such that legitimate web traffic intended for actual web users is slowed down and delayed. If this type of service becomes too slow, time sensitive information such as live video footage may be rendered entirely useless to legitimate end users. The botnet controller community features a constant and continuous struggle over who has the most bots, the highest overall bandwidth, and the most "high-quality" infected machines, like university, corporate, and even government machines.

II. RELATED WORK

Despite the fact that DNS tunneling is known for bypassing firewalls and encapsulating arbitrary data such as SSL traffic [4], Exploring DNS protocol as a practical C&C channel and identifying its limitations have not been scientifically studied. Various proof-of-concept botnet C&C systems via unconventional media exist, such as via bluetooth and social networks [2]. In comparison, our work is useful beyond the specific DNS-based communication channel studied in two aspects. They are present new quantitative techniques and evaluation regarding the detection and construction of general-purpose distributed stealthy communication systems, including temporal strategies for making stealthy communication and statistical content analysis. We give a practical technique that is useful in domain flux from the attacker's perspective, namely MC-based domain name generation.

DNS-based anomaly detection, Karasaridis et al. described the use of the Kullback-Leibler distance to measure byte distribution in DNS datagrams. Dagon proposed to quantify how anomalous the number of queries for each domain name during an hour in a day with Chebyshev's inequality and distance measures previously used for examining anomalous payloads. DNS-based anomaly detection approaches are presented for detecting botnet C&C activities. One method is to detect dynamic domain names whose query rates are abnormally high or temporally concentrated using outlier detection metrics such as Chebyshev's inequality. Our work describes stealthy DNS behaviors whose querying patterns are hard to distinguish with legitimate domains, which make the counting-based detection less effective.

Stone-Gross et al. observed the use of domain flux in Torpig botnet, where new communication domains are generated periodically and registered by the C&C server. Torpig bots communicated with the server over HTTP, after resolving the domain name. Patterns of fast-flux botnets are measured and analyzed in [1]. In comparison, we investigate the feasibility of solely DNS-based C&C, without requiring any additional web servers. The work in [8] utilizes machine learning techniques to identify domain names that are algorithmically generated. Though it remains unclear whether our MC-generated domain names can be experimentally distinguished from legitimate domain names by the techniques, we conjecture that the MC-generated domains would be difficult to distinguish from legitimate ones. The work in [6] describes 15 features that can be used to detect anomalous DNS traffic in wide area networks, including IPs, TTL values, temporal features. The stealth techniques on query pattern and domain name generation described in this work may help evade the machine-learning-based detection, showing the need for further research in this direction. Our work is complementary to host-based malware detection and prevention solutions, such as the cryptographic provenance verification technique [3].

III. BOTNET COMMUNICATION PROCESS

Botnet command-and-control (C&C) channel refers to the protocol used by bots and botmaster to communicate to each other, e.g., for bots to receive new attack commands and updates from botmaster, or to submit stolen data. A C&C channel for a botnet needs to be reliable, redundant, noncentralized, and easily disguised as legitimate traffic. Many botnet operators used the Internet Relay Chat protocol (IRC) or HTTP servers to pass information. Botnet operators constantly explore new stealthy communication mechanisms to evade detection. HTTP-based command and control is difficult to distinguish from legitimate web traffic. The feasibility of email as a stealthy botnet command and control protocol was studied by researchers. In this paper, we systematically investigate the feasibility of solely using Domain Name System (DNS) queries for botnet command and control. DNS provides a distributed infrastructure for storing, updating, and disseminating data that conveniently fits the need for a large-scale command and control system. The HTTP protocol is for the end-to-end communication between a client and a server. In comparison, DNS provides not only a means of communication between computers, but also systematic mechanisms for naming, locating, distributing, and caching resources with fault tolerance. These features of DNS may be utilized to fulfill a more effective command-and-control system than what HTTP servers may provide. The decentralized nature of DNSs with a series of redundant servers potentially provides an effective channel for covert communication of a large distributed system, including botnets. To play the devil's advocate, we focus on systematically analyzing the feasibility of a pure DNS-based C&C. Such a study has never been reported in the literature. Our C&C system is compatible with existing DNS infrastructure without enlisting any web or special-purpose servers. The DNS channel is aided by being a high-traffic channel such that data can be easily hidden. As virtually anyone can create and register their own domain names and DNS servers, it is a system that can be easily infiltrated by hackers and botnet operators.

DNS tunneling is a technique known for transmitting arbitrary data via DNS protocol, e.g., DNScat and DeNiSe. One application of DNS tunneling is to bypass firewalls, as both inbound and outbound DNS connections are usually allowed by organizational firewall rules. Because DNS is often overlooked in current security measures, it offers a C&C channel that is unimpeded. Because nearly all traffic requires DNS to translate domain names to IP addresses and back, simple firewall rules cannot easily be created without harming legitimate traffic. Recently, Dietrich et al. [5] reported Feederbot that used DNS as a communication channel for C&C traffic. However, Feederbot fails to utilize any distributed storage and query mechanisms offered by DNS. This botnet simply tunnels its command and control traffic by sending it in DNS format for the end-to-end communication between bots and the bot master. The domains used by them are not registered and cannot be resolved. We perform comprehensive experiments to evaluate the behaviors of proposed query strategies in terms of how quickly new commands are disseminated to a large number of bots. Our analysis utilizes a 4.6-GB two-month-long wireless network trace obtained from an organization. We conclude that the DNS-based botnet C&C channel is feasible, powerful, and difficult to detect and block.

IV. DNS QUERY MODELS AND ATTACKS

We describe an exponentially distributed query strategy and a piggybacking query strategy, both can be used to hide bot activities while communicating with a botmaster in a timely fashion. We provide an experimental evaluation on both query methods. Exponentially distributed query strategy. The Poisson process is previously believed to be a suitable model for representing stochastic processes, where arrivals are independent on each other, i.e., memoryless. DNS request arrivals are modeled by Poisson processes with exponential random variables with different rates λ. In our exponentially distributed query strategy, a bot probabilistically distributes DNS queries so that their intervals follow an exponential distribution with a parameterized arrival rate λ_b. Because of the memoryless feature of the model, the bot does not need to

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

store the previous communication history [7]. One simple way to implement this query strategy is as follows. They are bot sends a DNS query, an interval t by drawing from an exponential distribution with parameter λ_b and bot sleeps for t and repeats from Step 1. There is a tradeoff between being stealthy and communication efficiency. We study a bot's strategy in finding an optimal λ_b, given the host-wide DNS query rates.

Piggybacking query strategy. Many websites contain content from multiple independent domains due to third-party content delivery, advertisements, or content mashup. Therefore, multiple DNS queries are usually issued by a host with temporal proximity. The composition of domains is usually dynamic. The piggybacking query strategy leverages this fact. A bot passively listens on the host's DNS traffic or name-translation related function calls and sends DNS queries when legitimate DNS queries are being made. Thus, the bot's query is blended among a group of legitimate DNS queries.

In the piggybacking query strategy, a bot's communication with the controller is constrained by the host's activities. Therefore, we focus on analyzing its timeliness, in terms of the dissemination efficiency of new command and data. We define time-to-communicate (TTC), minimum TTC, and maximum TTC. Minimum TTC is a threshold aiming to prevent a bot from sending queries too frequently, whereas maximum TTC is a threshold for keeping the liveliness of the communication between the bot and the bot master in case of an inactive host. In this piggybacking mode, bots need to know when a legitimate DNS query is made. There are different approaches for obtaining this information. Traffic sniffing DNS server runs on port 53, an outgoing packet from the host to a destination IP on port 53 is an indication of an outbound DNS request. The bot may sniff the network traffic to identify the right moment to issue its DNS queries. The bot may either directly program with pcap library or call existing tools such as iftop, which uses pcap. This approach gives a cross platform and system-wide monitoring solution, which covers the DNS queries from any application. We have implemented this approach in Linux. With the pcap library, the traffic sniffing is relatively straightforward to realize with a small piece of code. We attach the code to host machine's network devices, which usually requires root privilege. In the code, function pcap_findalldevs() returns all available network devices, pcap_open_live() opens a network device, and pcap_loop() performs the traffic sniffing. The program needs root privilege to run. It filters all TCP and UDP packets, both of which are possible protocols for carrying DNS requests, and reports the time stamps whenever a DNS request is identified. It also distinguishes the bot's own DNS query from legitimate ones, so that piggybacking is only performed on legitimate DNS queries. This prototype demonstrates a feasible way for our proposed piggybacking query strategy.

Process hooking is a hook DNS related function. In Linux, bots can watch the calls for DNS-related APIs such as gethostbyname() function in libbind library. gethostbyname looks up all IP addresses associated with a host name and is implemented in the resolver library. One way of hooking into the API function is for bots to register a .so file to the LD_PRELOAD environmental variable, which may or may not require root. In the registered .so file, the target API function is replaced by the attacker's version which can notify the bot whenever this function is called. Similarly, in Windows, gives a solution, Detour, which has the ability to perform function interception and rewriting. Compared with traffic sniffing approach above, this method is less desirable, as it is process specific. The hooking is done in the process memory. Rogue library attackers may replace the network related shared libraries in the OS with an instrumented version so that the shared library is changed permanently on the hard disk. The rogue library is loaded by any application. It reports every DNS request to the bot. This approach is system wide, so the change is once and-for-all.

V. ISSUES ON BOTNET COMMUNICATION AND ATTACKS

Botnet controllers use stealthy messaging systems to set up large-scale command and control requests. A C&C channel for a botnet needs to be reliable, redundant, noncentralized and legitimate traffic. Domain Name Service (DNS) provides a distributed infrastructure for storing, updating and disseminating data. DNS is targeted as a stealthy botnet command-and-control channel. Malicious DNS activities are hide at the network level. Exponentially Distributed Query and Piggybacking Query attacks are detected using the markov chain analysis and statistical analysis mechanism. Probability distribution based analysis model is used to detect automatic domain flux attacks. DNS tunneling technique is

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

used for transmitting arbitrary data via DNS protocol. The following issues are identified in the current DNS Communication Schemes.

- ✓ DNS sensitive data access is not controlled
- ✓ Data leaks are not accurately detected
- ✓ User intention is required for anomaly detection
- ✓ Detection latency is high.

VI. PRIVACY PRESERVED DNS ATTACK DETECTION

The Domain Name Service based attack detection system is designed to handle command and control message process over DNS query values. The DNS query analysis is performed with statistical analysis. The system analyzes the DNS query values to identify the insertion of DDoS attack detection information. Machine learning approach is used to detect the DDoS attacks. The system is divided into five major modules. They are request observer, statistical analysis, Bayesian analysis, command and control request handler and security and privacy process. The request observer module is designed to collect service requests from the clients. Statistical analysis module is designed to detect DDoS attacks using statistical methods. Bayesian approach based attack detection mechanism uses the classification process. The command and control handler is designed to manage the DNS query based botnet communication activities. Security and privacy module is designed to provide security and privacy for DSN parameters.

6.1. Request Observer

Stream requests are collected from various clients. The server assigns session instances to new stream requests. The stream requests are processed by the web server. The responses are redirected to the clients.

6.2. Statistical Analysis

The statistical analysis model is used to detect the Service attacks. The request flow and its similarity are analyzed with frequency and interval values. The user session patterns are learned from the request flow analysis. Attack decisions are made with the support of DNS query format. The requests are assigned with normal or attack labels in the analysis.

6.3. Bayesian Analysis

The Bayesian analysis model is used to detect DNS query based attacks using machine learning approaches. The classification model is used for the request category identification process. The Bayesian classification algorithm is used for the attack detection process. The attacker requests are rejected by the service provider.

6.4. Command and Control Request Handler

The command and control requests are exchanged between the botmaster and bots. The DNS query values are used to exchange the command and control requests. The DNS query responses are verified with the domain name server details. Command and control instructions are dropped with reference to their category.

6.5. Security and Privacy

The DNS query values are submitted from the clients in the networks. The DNS responses are prepared by the domain name server. The user can add additional parameters to the DNS responses. The RSA algorithm is used to protect the DNS parameter values. Sensitive attribute values are protected with cryptography methods.

VII.CONCLUSION

The DNS query based attack detection scheme is enhanced to provide privacy preserved data traffic analysis. Automated anomaly detection is adapted to the system. Naiva Bayesian classification technique is integrated to the system. Small query analysis mechanism is integrated with the system. The system performs botnet communication detection and control operations. DDoS attack detection mechanism is included in the system. The system improves the detection accuracy with minimum latency. DNS parameter security is also provided by the system.

REFERENCES

[1] X. Hu and K.G. Shin, "Measurement and Analysis of Global IP-Usage Patterns of Fast-Flux Botnets," Proc. 30th Ann. Int'l Conf. Computer Comm., 2011.

[2] E. Kartaltepe, Sandhu, "Social Network-Based Botnet Command-and-Control: Emerging Threats and Countermeasures," Proc. Eighth Int'l Conf. Applied Cryptography and Network Security, 2010.

[3] K. Xu, H. Xiong and D. Yao, "Data-Provenance Verification for Secure Hosts," IEEE Trans. Dependable and Secure Computing, Mar.-Apr. 2012.

[4] DeNiSe, http://c0re.23.nu/c0de/snap/DeNiSe-snap-20021026.tar.gz, 2013.

[5] C.J. Dietrich, C. Rossow, F.C. Freiling, H. Bos, M. van Steen, and N. Pohlmann, "On Botnets that Use DNS for Command and Control," Proc. European Conf. Computer Network Defense, Sept. 2011.

[6] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, "Exposure: Finding Malicious Domains Using Passive DNS Analysis," Proc. 18th Ann. Network and Distributed System Security Symp. (NDSS), Feb. 2011.

[7] Kui Xu, Patrick Butler, Sudip Saha, and Danfeng (Daphne) Yao, "DNS for Massive-Scale Command and Control", IEEE Transactions On Dependable and Secure Computing, Vol. 10, No. 3, May/June 2013.

[8] S. Yadav, and S. Ranjan, "Detecting Algorithmically Generated Malicious Domain Names," Proc. 10th Ann. Conf. Internet Measurement, pp. 48-61, 2010.