www.ijecs.in

International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 12 December 2014, Page No. 9717-9720

Detecting and Preventing Distributed Denial of Service (DDOS) Attacks Using BOTNET Monitoring System

S.Manikandan [1], K. Manikanda Kumaran [2], S.Palanimurugan [3], S.Aravindan [4] and S.Praveen Kumar [5]

[1,2,3] Assistant Professor, Dept. of IT,
EGSPEC, Nagapattinam, Tamilnadu, India.
e-mailID: profmaninvp@gmail.com

[4,5] Assistant Professor, Dept. of CSE,
EGSPEC, Nagapattinam, Tamilnadu, India.
e-mailID: kkl.aravind@gmail.com

Abstract - Denial-of-Service (DoS) attacks pose a significant threat to the Internet today especially if they are distributed, i.e., launched simultaneously at a large number of systems. Reactive techniques that try to detect such an attack and throttle down malicious traffic prevail today but usually require an additional infrastructure to be really effective. In this paper we show that preventive mechanisms can be as effective with much less effort: We present an approach to (distributed) DoS attack prevention that is based on the observation that coordinated automated activity by many hosts needs a mechanism to remotely control them. To prevent such attacks, it is therefore possible to identify, infiltrate and analyze this remote control mechanism and to stop it in an automated fashion. We show that this method can be realized in the Internet by describing how we infiltrated and tracked distributed denial of service attacks using hybrid peer to peer botnets monitoring system.

Index: Botnet, DDOS, Honeypot, IDS and Mimic Flash Crowds

1. Introduction

Today's Internet connected networks are under permanent attack by intruders and automated attacks of worms. A variety of detection tools exist such as Intrusion Detection systems (IDS) and firewalls, but the main problem is that they only react on reconfigured and therefore known attacks. Botnets are an upcoming technology that can be used to detect and analyze network attacks. A Botnet is an apparently vulnerable system deployed to be hacked. An analysis of current Botnet approaches has been made and it has been evaluated in how far these approaches can contribute to the analyzation process. Some tests have shown that Botnets are exposed to lots of known attacks and noise that hide the valuable information about new attacks and vulnerabilities [1].

In this paper we present the design of an advanced hybrid peer-to-peer botnet. The focus of this is to analyze how far Botnet technology can be used to detect new, unknown attacks and this gathers information about the hackers that can be used to improve the security of our network. Botnets help to develop a reasoned, proactive response to a threat. Bot IDS can protect important files and directories on our hard disk no matter what file-system type they reside on, anybody include root can not change the files. Botnet IDS can also protect the important process from being killed.

1.1 Surviving Organized DDOS Attacks that Mimic Flash Crowds

Recent denial of service attacks are mounted by professionals using Botnets of tens of thousands of compromised machines. To circumvent detection, attackers are increasingly moving away from bandwidth floods to attacks that mimic the Web browsing behavior of a large number of clients, and target expensive higher-layer resources such as CPU, database and disk bandwidth. The resulting attacks are hard to defend against using standard techniques, as the malicious requests differ from the legitimate ones in intent but not in content. We present the design and implementation of Kill-Bots, a kernel extension to protect Web servers against DDoS attacks that masquerade as flash crowds. Kill-Bots provides authentication using graphical tests but is different from other systems that use graphical tests [2].

First, Kill-Bots uses an intermediate stage to identify the IP addresses that ignore the test, and persistently bombard the server with requests despite repeated failures at solving the tests. These machines are bots because their intent is to congest the server. Once these machines are identified, Kill-Bots blocks their requests, turns the graphical tests off, and allows access to legitimate users who are unable or unwilling to solve graphical tests. Second, Kill-Bots sends a test and checks the client's answer without allowing unauthenticated clients access to sockets, TCBs, and worker processes. Thus, it protects the authentication mechanism from being DDoSed. Third, Kill- Bots combines authentication with admission control. As a result, it improves performance, regardless of whether the server overload is caused by DDoS or a true Flash Crowd.

1.3 Paper Organization

The rest of the paper is organized as follows. Section 2 introduces system analysis and specifies the algorithms. Section 3 introduces the control communication architecture of the proposed botnet and botmasters. Section 4 describes data encryption/decryption standard and algorithm. Finally, conclude the paper in Section 5.

2. System Analysis

Traditionally, information security has been purely defensive. Firewalls, Intrusion Detection Systems, encryption; all of these mechanisms are used defensively to protect one's resources. A variety of detection tools exist such as Intrusion Detection systems (IDS) and firewalls, but the main problem is that they only react on reconfigured and therefore known attacks. In an existing system that will produce only the simulation result. BOTMASTER does not know the hackers IP address. This system can only run on single system. The primary purpose of a Honey net is to gather information about threats that exist [3].

A Honey net is a type of honey pot. Specifically, it is a high-interaction honey pot designed to capture extensive information on threats. High-interaction means a Honey net provides real systems, applications, and services for attackers to interact with. It is through this extensive interaction we gain information on threats, both external and internal to an organization. Proposed system can note the IP address of Hackers and can identify what type of file they want to access and what password and key was given by hackers to access the file. This system can produce the real time result. We can run it on more than one system without changing, and can run in single system too.

2.1 An algorithm for anomaly-based botnet detection

We present an anomaly-based algorithm for detecting IRC-based botnet meshes. The algorithm combines an IRC mesh detection component with a TCP scan detection heuristic called the TCP work weight. The IRC component produces two tuples, one for determining the IRC mesh based on IP channel names, and a sub-tuple which collects statistics (including the TCP work weight) on individual IRC hosts in channels. We sort the channels by the number of scanners producing a sorted list of potential botnets. This algorithm has been deployed in PSU's DMZ for over a year and has proven effective in reducing the number of botnet clients [4].

Data Control

Data Control is the containment of activity. It is what mitigates risk. By risk, we mean there is always the potential of an attacker using a Botnet to attack or harm non-Botnet systems. We want to make every effort possible to ensure that once an attacker is within our Botnet, they cannot accidentally or purposefully harm other non-Botnet systems. This is more challenging then it seems. First, we have to allow the attackers some degree of freedom to act. The more activity we allow the attackers to perform, the more we can potentially learn about them. However, the more freedom you allow an attacker, the more risk there is they will circumvent Data Control and harm other non-Botnet systems.

Data Capture

Data Capture is the monitoring and logging of all of the black hat's activities within the Botnet. It is this captured data that is then analyzed to learn the tools, tactics, and motives of members of the black hat community. The challenge is to capture as much data as possible, without the blackhat detecting the process. As with Data Control, one of the primary lessons learned for Data Capture has been the use of layers. It is critical to use multiple mechanisms for capturing activity. Not only does the combination of layers help piece together all of the attacker's actions, but it prevents having a single point of failure.

Data Collection

Data Collection is a third requirement, but this only applies to organizations that have multiple Botnets in distributed environments. Many organizations will have only one single Botnet, so all they need to do is both Control and Capture data. However, organizations that have multiple Botnets logically or physically distributed around the world, such as the Botnet Research Alliance have to collect all of the captured data and store it in a central location.

3. Botnet Monitoring by its Botmaster

Another major challenge in botnet design is making sure that a botnet is difficult to monitor by defenders, but at the same time, easily monitored by its botmaster. With detailed botnet information, a botmaster could (1). Conduct attacks more effectively according to the bot population, distribution, on/off status, IP address types, etc; (2). Keep tighter control over the botnet when facing various counterattacks from defenders [fig 1]. In this section, we present a simple but effective way for botmasters to monitor their botnets whenever they want, and at the same time, resist being monitored by others.

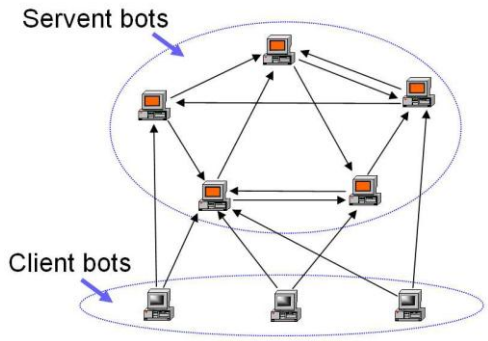

Fig 1. Command and control architecture of the proposed hybrid P2P botnet

3.1 Botmaster Implementation

1. Use a popular Internet service, such as HTTP or Email, for report to a sensor. The sensor is chosen such that it normally provides such a service to avoid exhibiting abnormal network traffic.
2. Use several sensor machines instead of a single sensor.
3. Select sensor hosts that are harder to be shut down or monitored, for example, compromised machines in other countries with minimum Internet security and International collaboration. Manually verify the selected sensor machines are not honeypots.
4. Wipe out the hard drive on a sensor host immediately after retrieving the report data.
5. Specify expiration time in report command to prevent any bot exposing itself after that time.
6. Issue another command to the botnet to cancel the previous report command once the botmaster knows that the sensor host has been captured by defenders.

To make sure that a constructed botnet is connected, the initial set of bots should contain some servent bots whose IP addresses are in the peer list in every initial bot. Suppose the size of peer list in each bot is configured to be *M*. As a bot program propagates, the peer list in each bot is constructed according to the following procedure (fig 1):

New infection: Bot A passes its peer list to a vulnerable host B when compromising it. if A is a servent bot, B adds A into its peer list (by randomly replacing one entry if its peer list is full). If A knows that B is a servent bot (A may not be aware of B's identity, for example, when B is compromised by an email virus sent from A), A adds B into its peer list in the same way.

Reinfection: If reinfection is possible and bot A reinfects bot B, bot B will then replace R ($R \leq M-1$) randomlyselected bots in its peer list with R bots from the peer list provided by A. Again, bot A and B will add each other into their respective peer lists if the other one is a servent bot as explained in the above "new infection" procedure.

Peer-list updating: After a botnet spreads out for a while, a botmaster issues a report command to obtain the information of all currently available servent bots. These servent bots are called peer-list updating servent bots. Then, the botmaster issues another command, called update command, enabling all bots to obtain an updated peer list from a specified sensor host. The sensor host randomly chooses M servent bots to compose an updated peer list, then sends it back to each requested bot.

4. Data Encryption / Decryption

The Blow fish involves replacing each letter of the alphabet with the letter standing k places further down the alphabet.

Encryption:

Blowfish is a Feistel network consisting of 16 rounds. The input is a 64-bit data element, x [Table 1].

Decryption

It is exactly the same as encryption, except that P1, P2,..., P18 are used in the reverse order. This algorithm used to encrypt the all the data before going to send to the user. Using the private key k it is decrypted on the end user side. The user who knows the private key can only decrypt the data.

Divide x into two 32-bit halves: xL, xR
For i = 1 to 16:
xL = xL XOR Pi
xR = F(xL) XOR xR
Swap xL and xR
Swap xL and xR (Undo the last swap)
xR = xR XOR P17
xL = xL XOR P18
Recombine xL and xR
Function F (see Figure 2):
Divide xL into four eight-bit quarters: a, b, c, and d
$F(xL) = ((S1,a + S2,b \bmod 2^{32}) \text{ XOR } S3,c) + S4,d \bmod 2^{32}$

Thus the botnet robustness metric $C(p)$ is:
$C(p) = 1, < R$ bots are removed
$0, \geq R$ bots are removed

The botnet will be shut down if all R C&C server bots are removed, which makes it much less robust than the proposed P2P botnet.

Table 1. Encryption and Decryption algorithm.

4.2 Logs and Alert System

This provide alert message to the administrator. If hacker entered into the network then the firewall detecting that hacker and immediately it tell to the honey pot about the hacker. Suddenly the Honey Pot monitoring the hacker activities. Before that it gives the alert message to the log/alert server where the administrator is sited. Then the administrator can watch the hacker's motivation and activities.

4.3 Botnet Monitoring Based on Spying Honeypots

If a botnet cannot effectively detect honeypots, defenders could let their honeypots join botnets and monitor botnet activities. Based on honeypot bots, defenders may be able to obtain the plain text of commands issued by a botmaster. Once the meaning of the commands is understood, defenders are able to: (1). Quickly find the sensor machines used by a botmaster in report commands. If a sensor machine can be captured by defenders before the collected information on it is erased by its botmaster, they might be able to obtain detailed information of the entire botnet; (2). Know the target in an attack command so that they could implement corresponding countermeasures quickly right the actual attack begins.

The proposed hybrid P2P botnet represents only a specific P2P botnet design. In reality, botmasters may come up with some other types of P2P botnet designs. However, we believe this research is still meaningful to security community. The proposed design is practical and can be implemented by botmasters with little engineering complexities. Botmasters will come with a similar design sooner or later, and we must be well prepared for such an attack, or a similar attack, before it happens.

5. Conclusion

Due to their potential for illicit financial gain, "botnets" have become popular among Internet attackers in recent years. As security defenders build more honeypot-based detection and defense systems, attackers will find ways to avoid honeypot traps in their botnets. Attackers can use software or hardware specific codes to detect the honeypot virtual environment, but they can also rely on a more general principle to detect botnet: security professionals using botnet have liability constraints such that their botnet cannot be configured in a way that would allow them to send out real malicious attacks or too many malicious attacks. In this paper, we introduced various means by which attackers could detect botnet in their constructed botnets based on this principle. Botnet research and deployment still has significant value for the security community, but we hope this paper will remind honeypot researchers of the importance of studying ways to build covert botnet and the limitation in deploying botnet in security defense.

The current popular research focused on finding effective honeypot-based detection and defense approaches will be for naught if botnet remain as easily detectible as they are presently. The P2P botnet is much harder for security professionals to track. There is no centralized bot controller they can monitor. A honeypot bot in the botnet can only monitor a very small portion of the entire botnet. The attacker still has control over the remaining P2P botnet, even if the remaining botnet is broken into many separated smaller ones. Attackers also do not need to spend money buying Dynamic DNS services. Therefore, we believe more P2P botnets will be created in the future.

6. References

[1] S. Kandula, D. Katabi, M. Jacob, and A. Berger, "Botz-4-sale: Surviving organized ddos attacks that mimic flash crowds," in 2nd Symposium on Networked Systems Design and Implementation (NSDI), May 2005.

[2] C. T. News, "Expert: Botnets No. 1 emerging Internet threat," 2006, http://www.cnn.com/2006/TECH/internet/01/31/furst/.

[3] F. Freiling, T. Holz, and G. Wicherski, "Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks," CS Dept. of RWTH Aachen University, Tech. Rep. AIB-2005-07, April 2005.

[4] D. Dagon, C. Zou, and W. Lee, "Modeling botnet propagation using time zones," in Proceedings of 13th Annual Network and Distributed System Security Symposium (NDSS), Feburary 2006, pp. 235–249.

[5] A. Ramachandran, N. Feamster, and D. Dagon, "Revealing botnet membership using dnsbl counter-intelligence," in USENIX 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI 06), June 2006.

[6] E. Cooke, F. Jahanian, and D. McPherson, "The zombie roundup: Understanding, detecting, and disrupting botnets," in Proceedings of SRUTI: Steps to Reducing Unwanted Traffic on the Internet, July 2005.

[7] J. R. Binkley and S. Singh, "An algorithm for anomaly-based botnet detection," in USENIX 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI 06), June 2006.

[8] I. Arce and E. Levy, "An analysis of the slapper worm," IEEE Security & Privacy Magazine, Jan.-Feb. 2003.

Authors Profile

S. Manikandan received the **B.Tech,** degree in Information Technology with Distinction from the EGS Pillay Engineering College, Nagapattinam, Anna University, Chennai, India, in 2010 and received **M.E.** Computer Science and Engineering with University first and Honors in Annamalai University, Annamalai Nagar India, in 2012. Currently he is working Assistant Professor in EGS Pillay Engineering College, Nagapattinam, India. His research interest includes Colud Computing, Soft computing, Networking, Network Security and Pervasive Computing.