Modeling Advanced Botnet and Detecting Botmasters

Anitha Lakshmi V
M.Tech. Computer Science
Dr.MGR University
Madhuravayal, TamilNadu
E-mail: anithalakshmi83@gmail.com

Abstract - A "botnet" consists of a network of compromised computers controlled by an attacker ("botmaster"). Recently, botnets have become the root cause of many Internet attacks. To be well prepared for future attacks, it is not enough to study how to detect and defend against the botnets that have appeared in the past. More importantly, we should study advanced botnet designs that could be developed by botmasters in the near future. In this paper, we present the design of an advanced hybrid peer-to-peer botnet. Compared with current botnets, the proposed botnet is harder to be shut down, monitored, and hijacked. It provides robust network connectivity, individualized encryption and control traffic dispersion, limited botnet exposure by each bot, and easy monitoring and recovery by its botmaster. In the end, we suggest and analyze several possible defenses against this advanced botnet.

Keywords - *Botnet, peer-to-peer, robustness, honeypot.*

I. INTRODUCTION

In the last several years, Internet malware attacks have evolved into better-organized and more profit-centered endeavors. E-mail spam, extortion through denial-of-service attacks [1], and click fraud [2] represent a few examples of this emerging trend. "Botnets" are a root cause of these problems [3], [4], [5]. A "botnet" consists of a network of compromised computers ("bots") connected to the Internet that is controlled by a remote attacker ("botmaster") [5], [6]. Since a botmaster could scatter attack tasks over hundreds or even tens of thousands of computers distributed across the Internet, the enormous cumulative bandwidth and large number of attack sources make botnet-based attacks extremely dangerous and hard to defend against. Compared to other Internet malware, the unique feature of a botnet lies in its control communication network.

Most botnets that have appeared until now have had a common centralized architecture. That is, bots in the botnet connect directly to some special hosts (called "command-and-control" servers, or "C&C" servers). These C&C servers receive commands from their botmaster and forward them to the other bots in the network. From now on, we will call a botnet with such a control communication architecture a "C&C botnet." Fig. 1 shows the basic control communication architecture for a typical C&C botnet (in reality, a C&C botnet usually has more than two C&C servers). Arrows represent the directions of network connections. As botnet-based attacks become popular and dangerous, security researchers have studied how to detect, monitor, and defend against them [1], [3], [4], [5], [6], [7].

Most of the current research has focused upon the C&C botnets that have appeared in the past, especially Internet Relay Chat (IRC)- based botnets. It is necessary to conduct such research in order to deal with the threat we are facing today. However, it is equally important to conduct research on advanced botnet designs that could be developed by attackers in the near future. Otherwise, we will remain susceptible to the next generation of Internet malware attacks. From a botmaster's perspective, the C&C servers are the fundamental weak points in current botnet architectures. First, a botmaster will lose control of her botnet once the limited number of C&C servers are shut down by defenders. Second, defenders could easily obtain the identities (e.g., IP addresses) of all C&C servers based on their service traffic to a large number of bots [7], or simply from one single captured bot (which contains the list of C&C servers). Third, an entire botnet may be exposed once a C&C server in the botnet is hijacked or captured by defenders [4]. As network security practitioners put more resources and effort into defending against botnet attacks, hackers will develop and deploy the next generation of botnets with a different control architecture.

The proposed hybrid P2P botnet has the following features:

- The botnet requires no bootstrap procedure.
- The botnet communicates via the peer list contained in each bot. However, unlike Slapper [8], each bot has a fixed and limited size peer list and does not reveal its peer list to other bots. In this way, when a bot is captured by defenders, only the limited number of bots in its peer list are exposed.
- A botmaster could easily monitor the entire botnet by issuing a report command. This command instructs all (or partial) bots to report to a compromised machine (which is called a sensor host) that is controlled by the botmaster. The IP address of the sensor host, which is specified in the report command, will change every time a report command is issued to prevent defenders from capturing or blocking the sensor host beforehand.
- After collecting information about the botnet through the above report command, a botmaster, if she thinks necessary, could issue an update command to actively let all bots contact a sensor host to update their peer lists. This effectively updates the botnet topology such that it has a balanced and robust connectivity, and/or reconnects a broken botnet.
- Only bots with static global IP addresses that are accessible from the Internet are candidates for being in peer lists (they are called servent bots according to P2P terminologies [12] since they behave with both client and server features). This design ensures that the peer list in each bot has a long lifetime.
- Each servent bot listens on a self-determined service port for incoming connections from other bots and uses a self-generated symmetric encryption key for incoming traffic. This individualized encryption and individualized service port design makes it very hard for the botnet to be detected through network flow analysis of the botnet communication traffic.

II. EXISTING METHODOLOGY

A. Current P2P Botnets and Their Weaknesses

Considering the above weaknesses inherent to the centralized architecture of current C&C botnets, it is a natural strategy for botmasters to design a peer-to-peer (P2P) control mechanism into their botnets. In the last several years, botnets such as Slapper [8], Sinit [9], Phatbot [10], and Nugache [11] have implemented different kinds of P2P control architectures. They have shown several advanced designs. For example, some of them have removed the "bootstrap" process used in common P2P protocols.1 Sinit uses public key cryptography for update authentication [9]. Nugache attempts to thwart detection by implementing an encrypted/obsfucated control channel [11]. Nevertheless, simply migrating available P2P protocols will not generate a sound botnet, and the P2P designs used by several botnets in the past are not mature and have many weaknesses. To remove bootstrap procedure, a Sinit bot uses random probing to find other Sinit bots to theoretical analysis of the lifetimes of both protocols and expressions for performance with respect to routing.

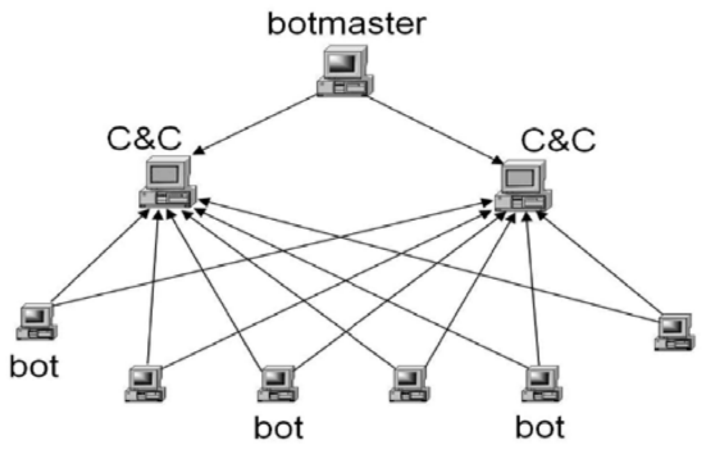

Figure 1: C&C Architecture of a C&C Botnet.

III. PROPOSED METHODOLOGY

A. Two Classes of Bots

The bots in the proposed P2P botnet are classified into two groups. The first group contains bots that have static, nonprivate IP addresses and are accessible from the global Internet. Bots in the first group are called servent bots since they behave as both clients and servers.2 The second group contains the remaining bots, including 1) bots with dynamically allocated IP addresses, 2) bots with private IP addresses, and 3) bots behind firewalls such that they cannot be connected from the global Internet. The second group of bots is called client bots since they will not accept incoming connections. Only servent bots are candidates in peer lists. All bots, including both client bots and servent bots, actively contact the servent bots in their peer lists to retrieve commands. Because servent bots normally do not change their IP addresses, this design increases the network stability of a botnet. This bot classification will become more important in the future as a larger proportion of computers will sit behind firewall, or use "Dynamic Host Configuration Protocol" (DHCP) or private IP addresses due to shortage of IP space. A bot could easily determine the type of IP address used by its host machine. For example, on a Windows machine, a bot could run the command "ipconfig /all." Not all bots with static global IP addresses are qualified to be servent bots—some of them may stay behind firewall, inaccessible from the global Internet. A botmaster could rely on the collaboration between bots to determine such bots. For example, a bot runs its server program and requests the servent bots in its peer list to initiate connections to its service port. If the bot could receive such test connections, it labels itself as a servent bot. Otherwise, it labels itself as a client bot.

Botnet Command and Control Architecture

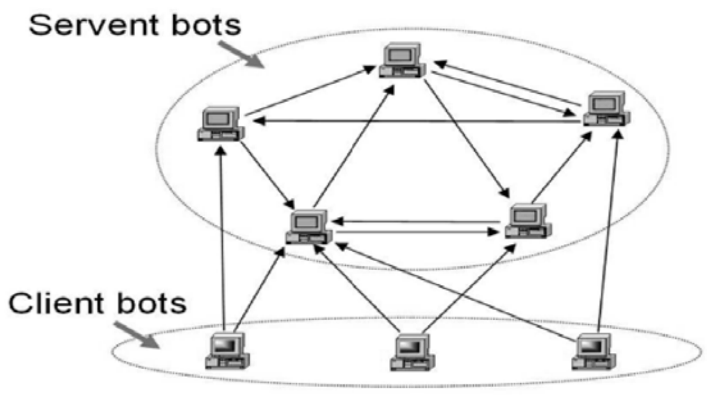

Figure 2:C&C Architecture of the Proposed Hybrid P2P Botnet.

Fig. 2 illustrates the C&C architecture of the proposed botnet. The illustrative botnet shown in this figure has five servent bots and three client bots. The peer list size is two (i.e., each bot's peer list contains the IP addresses of two servent bots). An arrow from bot A to bot B represents bot A initiating a connection to bot B. This figure shows that a big cloud of servent bots interconnect with each other—they form the backbone of the control communication network of a botnet. A botmaster injects her commands through any bot(s) in the botnet. Both client and servent bots periodically connect to the servent bots in their peer lists in order to retrieve commands issued by their botmaster. When a bot receives a new command that it has never seen before (e.g., each command has a unique ID), it immediately forwards the command to all servent bots in its peer list. In addition, if itself is a servent bot, it will also forward the command to any bots connecting to it.

IV. BOTNET DETECTION AND MONITORING WITHOUT HONEYPOTS

The previous section shows that we can propose many effective botnet monitoring approaches based on honeypot techniques. However, as honeypot-based defense systems gradually become popular and widely deployed, botmasters will inevitably develop their botnets to detect honeypots. For this reason, we propose botnet detection and monitoring approaches that do not rely on honeypots.

A. Monitoring Traffic to Botnet Sensor

A possible weakness point of the proposed botnet is its centralized monitoring sensor. If defenders have set up a good traffic logging system, it is possible that they could capture the traffic to a botnet sensor. We call such a monitoring system as a botnet sensor monitor. Even though defenders maynot be able to capture a botnet sensor before its botmaster destroys the sensor (after completing botmaster's monitoring task), they still could use the captured traffic log to figure out the IP addresses of botswhocontacted the sensor in the past. In this way, defenders could get a relatively complete picture of a botnet.

B. Detecting and Monitoring Servent Bots

In the proposed hybrid P2P botnet, servent bots, especially those used in the peer-list updating procedure, are the backbone of a botnet. Fig. 4 shows that each servent bot used in the peer-list updating will serve 300 to 500 bots. If a nonserver host is infected and serves as one of these servent bots, the host is relatively easy to be spotted by defenders due to the huge increase of traffic in and out of this host. When the number of servent bots compared to the total botnet population decreases, each of these servent bots must serve a larger number of bots and, hence, is easier to be detected by defenders. A simple statistical analysis can show this relationship. Denote the number of client bots served by a servent bot as D. For the proposed botnet, D is a random variable. Suppose the peer-list updating procedure is conducted after a botnet finishes its propagation, then all servent bots are used in the updating procedure; and hence, they have evenly distributed connection degrees. Following the same notations as in previous analysis, K is the number of servent bots in a botnet, I is the botnet size, and M is the peer list size. It is not hard to derive the distribution of D. In the peerlist updating procedure, each client bot is given a randomly chosen peer list by the updating sensor. For any specific servent bot, each client bot has an equal and small probability M=K to connect to the servent bot.

V. BOTNET CONSTRUCTION

I. Basic Construction Procedure

A natural way to build peer lists is to construct them as a botnet propagates. To make sure

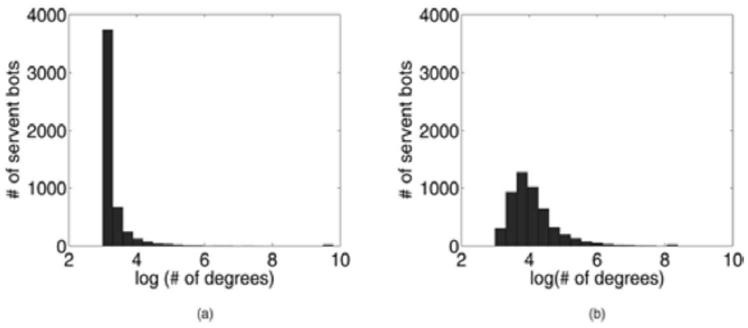

Figure 3:Servent Bot Degree Distribution

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 onfigured to be M. As a bot program propagates, the peer list in each bot is constructed according to the following procedure: .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 e-mail 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 randomly selected bots in its peer list with R bots from the peer list provided by A. Again, bots A and B will add each other into their respective peer lists if the other one is a servent bot as explained in the Figure 3,Servent bot degree distribution (construct botnet via "new infection" and "reinfection" procedure only). (a) Vulnerable population 500,000. (b) Vulnerable population 20,000.above “new infection” procedure. Fig. 3 shows the degree distribution for serventbots (client bots always have a greeM, equal to the size of peer list) after the botnet has accumulated 20,000 members.

II. Advanced Construction Procedure

One intuitive way to improve the network connectivity would be letting bots keep exchanging and updating their peer lists frequently. However, such a design makes it very easy for defenders to obtain the identities of all servent bots, if one or several bots are captured by defenders. With the detailed botnet information, a botmaster could easily update the peer list in each bot to have a strong and balanced connectivity. The added new procedure is 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. Fig. 4 shows the degree distribution for servent bots (client bots always have a degree of M) when a botnet uses all three construction methods.

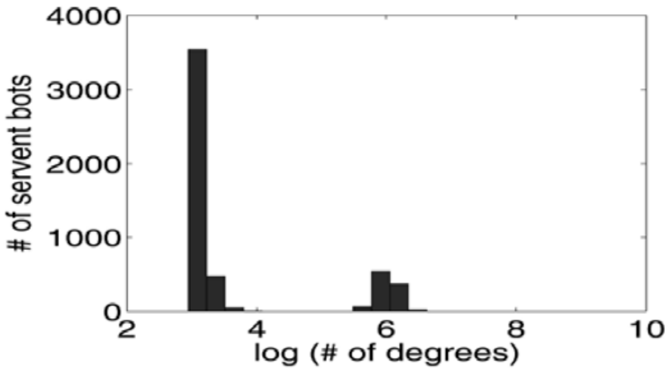

Figure 4:ServentBot Degree Distribution (Constructed via Infection and Peer-List Updating).

VI. CONCLUSION

To be well prepared for future botnet attacks, we should study advanced botnet attack techniques that could be developed by botmasters in the near future. In this paper, we present the design of an advanced hybrid P2P botnet. Compared with current botnets, the proposed one is harder to be monitored, and much harder to be shut down. It provides robust network connectivity, individualized encryption and control traffic dispersion, limited botnet exposure by each captured bot, and easy monitoring and recovery by its botmaster. To defend against such and advanced botnet, we point out that honeypots may play an important role. We should, therefore, invest more research into determining how to deploy honeypots efficiently and avoid their exposure to botnets and botmasters.

REFERENCES

1. S. Kandula, D. Katabi, M. Jacob, and A. Berger, "Botz-4-Sale: Surviving Organized DDOS Attacks That Mimic Flash Crowds,"Proc. Second Symp. Networked Systems Design and Implementation (NSDI '05), May 2005.
2. C.T.News, Expert: Botnets No. 1 Emerging Internet Threat, http:// www.cnn.com/2006 / TECH/internet/01/31/furst/, 2006.
3. F. Freiling, T. Holz, and G. Wicherski, "Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of- Service Attacks," Technical Report AIB-2005-07, CS Dept. RWTH Aachen Univ., Apr. 2005.
4. D. Dagon, C. Zou, and W. Lee, "Modeling Botnet Propagation Using Time Zones," Proc. 13th Ann. Network and Distributed System Security Symp. (NDSS '06), pp. 235-249, Feb. 2006.
5. Ramachandran, N. Feamster, and D. Dagon, "Revealing Botnet Membership Using DNSBL Counter-Intelligence," Proc. USENIX Second Workshop 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," Proc. USENIX Workshop Steps to Reducing Unwanted Traffic on the Internet (SRUTI '05), July 2005.
7. J.R. Binkley and S. Singh, "An Algorithm for Anomaly-Based Botnet Detection," Proc. USENIX Second Workshop Steps to Reducing Unwanted Traffic on the Internet (SRUTI '06), June 2006.
8. Arce and E. Levy, "An Analysis of the Slapper Worm," IEEE Security & Privacy Magazine, vol. 1, no. 1, pp. 82-87, Jan.-Feb. 2003.
9. Sinit P2P Trojan Analysis, http://www.lurhq.com/sinit.html, 2008.
10. Phatbot Trojan Analysis, http://www.lurhq.com/phatbot.html, 2008.
11. R.Lemos, Bot Software Looks to Improve Peerage, http:// www. Securityfocus. Com/news/11390, May 2006.
12. Servent, http://en.wikipedia.org/wiki/Servent, 2008.
13. R.Puri, Bots & Botnet: An Overview, http://www.sans.org/rr/ whitepapers/malicious/ 1299.php, 2003.
14. McCarty, "Botnets: Big and Bigger," IEEE Security & Privacy Magazine, vol. 1, no. 4, pp. 87-90, July-Aug. 2003.
15. P. Barford and V. Yegneswaran, An Inside Look at Botnets, to appear in Series: Advances in Information Security. Springer, 2006.
16. H. Project, Know Your Enemy: Tracking Botnets, http:// www.Honeynet.org/papers/bots/, 2005.
17. F. Monrose, "Longitudinal Analysis of Botnet Dynamics," ARO/DARPA/DHS Special Workshop Botnet, 2006.