

Detecting P2P-Controlled Bots on the Host

Antti Nummipuro
Helsinki University of Technology
anummipu # cc.hut.fi

Abstract

Storm Worm is a trojan that uses a Peer-to-Peer (P2P) protocol as a command and control channel of a botnet, which it forms. These botnets have, for example, been used in Distributed Denial of Service (DDoS) attacks. There have been studies on detecting botnet command and control channels of traditional botnets at host level. This paper investigates if it is possible to apply some of these techniques to detect peer-to-peer botnets.

KEYWORDS: botnet, p2p, peer-to-peer, detecting, ddos, trojan, host

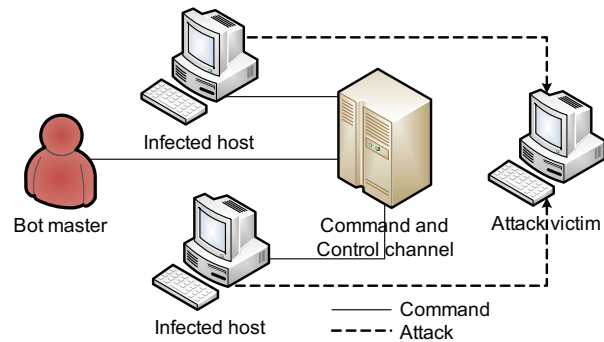


Figure 1: Main components of a botnet

1 Introduction

Botnets have mainly been used for the following criminal activities [12]:

- massive Distributed Denial of Service (DDoS) attacks
- spam emails for phishing, malware spreading, and advertising
- identity theft, that is collecting personal data for fraud
- blackmailing - you'll get a DDoS attack if you don't pay up
- click fraud

Indeed, the botnets are among the most important threats to the security of the Internet [3].

Main components of a botnet are bot master, infected hosts and command and control channel, see Figure 1. Hosts are infected by the bot by various ways, including malicious email attachments and scanning for vulnerabilities. The hosts form a network in which communication is done by using the command and control channel. The bot master can control the network via it. For example, he can initiate an attack or gather some data from the hosts. [12] [16]

Traditional botnets rely on Internet Relay Chat (IRC) for command and control. There are only few IRC servers where bots connect and the traffic might not even be encrypted. The IRC is an efficient way of communication, but it has a central point of failure [3]. The bot master loses the control over the botnet if the IRC servers, that are usually compromised computers, are blocked or cleaned up. The bots can also be detected at hosts by the traffic to the servers.

It is more difficult to shut down, monitor or hijack peer-to-peer botnets [18]. For example, Storm Worm has a hard coded list of over 100 peers that it can connect to get to the command and control channel [15]. If just one of them is working, the bot gets to the command and control network. Traffic can be distributed to many hosts. Recently a variation, that uses encryption, of Storm Worm has been found [14].

In a peer-to-peer network any host, i.e. peer, can act as both a client and a server. Generally, a peer connects to some already networked peers to get to the network. Then the peer can communicate with any peer that is in the network. The other peers route messages to not directly connected destinations. [10]

The Storm Worm uses the eDonkey/Overnet protocol, which is a decentralized peer-to-peer protocol [15]. It can mostly bypass firewalls if there's some unfirewalled peers. It is usually used for sharing large files. However, its importance to peer-to-peer file sharing is declined since the company behind it was forced to take down the eDonkey/Overnet resources, because of the legal actions from the RIAA [1]. It has not completely died as it is decentralized.

Figure 2 shows an example of how a peer-to-peer botnet works. Each bot keeps some connections to the other bots of the botnet. A new bot must know some addresses of the botnet to connect there. When a host is infected with a bot, it can contain a list of some bots of the botnet. After the bot is in the botnet it can update the list. The peer-to-peer network is the command and control channel of the bots. It can deliver messages from a single bot to other bots.

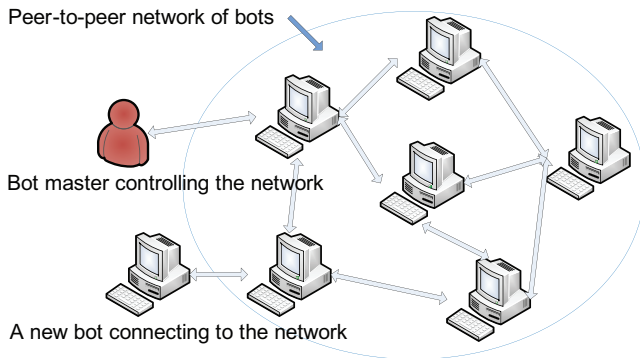


Figure 2: Example of peer-to-peer botnet architecture

Peer-to-peer networking is not going to vanish in the near future. Its common use, file sharing, is very popular. A single peer-to-peer file sharing web site has had over 3 billion downloads and it is in the Alexa top 100 list of most visited web sites of the Internet [17]. There is also increasing number of fully legitimate applications [13]. Even Microsoft Windows Vista includes peer-to-peer components [9].

2 Solutions

This section provides solutions to the peer-to-peer botnet problem described in Section 1. The solutions are based on tracking data from the network (Section 2.1), analyzing the data from the network (Section 2.2), and the approach of antivirus and behavior blocking (Section 2.3).

2.1 Tracking Data Received over the Network

The bot master sends a command over the command and control network and each bot independently executes the command. Many of the commands invoke operating system (OS) services on the host. Stinson et al. [16] have tested that this remote control behavior differentiates malicious bots from benign programs, like a web browser. They have implemented a prototype, BotSwat, that can monitor execution of any Windows XP or 2000 application.

BotSwat interposes on the run-time calls that a Windows process makes by hooking. The run-time calls can be Windows system calls (e.g., connect or file open) or other library calls (e.g., decrypt some data using a cryptographic library).

If the process uses data received from the network, the received data is considered tainted. The tainted data is tracked when it propagates via library calls to other memory regions.

They have selected specific gate functions to track. The gate functions are system calls likely to be used in malicious bot activity. The BotSwat is based on detecting if a gate function is called with parameters from the tainted memory.

While it is possible to utilize techniques that hide the path of the tracked data (e.g., out-of-band memory copies), they also perform content-based tainting. It considers a memory region tainted if it contains identical string to a known tainted string. That makes it harder to evade the detection.

The tested bot commands were implemented in significantly different ways, but there are similarities in the response to external control. That allows a single approach to detect them. The generic approach of BotSwat is claimed to not rely on specific command and control protocol or botnet structure, it should be able to detect even encrypted peer-to-peer bot command and control activity.

Hooking is the basis of tracking the data. When a Windows application requests for information, it calls a Windows API function. The request is eventually forwarded to the Windows kernel that processes it and sends results back via the same route it came, the execution path. Hooking is diverting the execution path to a function that access the request or the results of it [11]. Next paragraph gives an example of simple and widely used hooking.

The idea of System Service Table (SST) hooking is illustrated in Figure 3 [5] modified by the author. The SST is a table of pointers that contains the addresses of the Windows kernel functions, which implement the system functions. An address in the SST can be replaced to point to a hook function. When a hooked system function is called, the hook function intercepts it and has access to its parameters and result values.

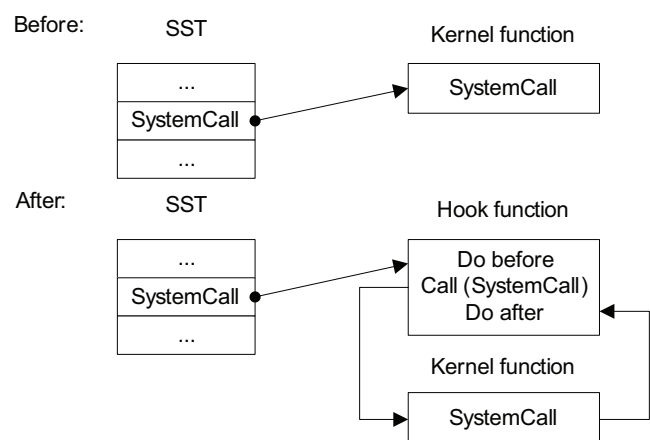


Figure 3: System Service Table hooking

SST is a type of hooking, but the main principle of getting into the execution path is the same with the all types.

2.2 Analyzing Data Received over the Network

A major mechanism used by bots to spread is to scan for vulnerabilities [18]. The vulnerabilities are in the software that a host is running and they can be exploited by a specially crafted network packets. The bot can be detected if it uses a known exploit method.

Denial of service (DoS) attacks could be detected by network activity. DoS attack generates a large number of similar packets to the attack target. That is quite unusual to normal network traffic a host makes. [2]

Also the command and control traffic could be detected by characteristics of network packets. Matt Jonkman has written some signatures based on network packets' size and frequency to detect Storm Worm in a generic way [4]. That is prone to false alarms due to not verifying the content.

2.3 Anti-virus and Behavior Blocking

Traditional antivirus approach is based on signatures of known malware, like viruses, bots and trojans. The detection is done by analyzing files accessed by the operating system against the signatures. Those are added to the antivirus software via updates, which can even be daily. Antivirus can clean infections and block new infections of the known malware. [8]

When Microsoft added the signatures of the Storm Worm to its Malicious Software Removal Tool (MSRT) on September 2007, it cleaned out a large number of infected machines [6]. However, the Storm Worm first appeared on November 2006 [15]. Either up-to-date antivirus software wasn't used largely enough or the software deployed on the field failed to clean the Storm Worm for some reason. The size of Storm Worm botnet seemed to peak in July 2007 [7].

Behavior blocking is done by a software that runs on the host [8]. When the software detects that a new program is trying to do something that is against the predefined policies (e.g., use the network), it asks the user to allow or deny the program to do that. In companies, the decision is usually made by administrators. They predefine the benign software and decide what to do with the unknown programs.

Behavior blocking is efficient against even unknown malware, if the user knows what not to allow. It does not require rapid signature updates, but it might be tricky to separate good and bad behavior.

3 Analysis

The nature of peer-to-peer combined to encryption and benign usage of peer-to-peer significantly limits the ability to use methods that are based on analyzing the contents of command and control channel's network traffic. While the characteristics of traditional, over the IRC, command and control channels allow the easier detection.

Home users expect to be able to use peer-to-peer applications, so all peer-to-peer traffic cannot be blocked from consumer Internet access - it would not be an easy task either as many peer-to-peer implementations are designed to overcome firewalls.

As peer-to-peer traffic is not being blocked, malicious traffic have to be detected from benign traffic. There are some approaches to this, but they are not reliable, as encryption makes it impossible to undoubtedly identify, that the traffic is what it is suspected to be.

That leads to a situation in which it is easier to analyse the botnet on the infected host. For that, there are the BotSwat, the traditional antivirus and the behavior blocking approaches.

The approach of BotSwat seems to be suitable solution. It does not matter whether the data is received from an IRC server or a perfect peer-to-peer network, it still must be processed on the host to do anything. The remote control functionality of botnets makes them behave distinctly from the usual benign applications. That is why it is also likely to detect new bot families and variations with this method.

The content-based tainting of BotSwat might make a large performance hit if it is used thoroughly. On the other hand, by not checking everything it is easier to evade the detection. Also BotSwat can make some false positives, but it should be less likely than with behavior blocking.

The antivirus approach is quite efficient against well known malware. Antivirus can clean infections and block new infections of the known malware. Currently the size of the Storm Worm botnet is a fraction of its peak magnitude, but it took more than half a year since the first appearance.

Behavior blocking can prevent the bots from making bad actions on the host after the infection. That still requires the user to deny the bot when a behavior blocking software asks what to do.

Regardless of the encrypted peer-to-peer control channel, the network based detection could be suitable to complement the host based methods. For example, Matt Jonkman's method could help to decide which network data is the most important to track at the host. Also a bot can be detected by the network packets of the attacks it does.

When an attack is known already, network based detection could block incoming attacks that would otherwise infect the host. Deploying the attack signature is usually faster than patching the vulnerabilities. The antivirus and firewall signatures get even daily updates, as the software patches might be monthly.

Table 1 summarizes the pros and cons of each method discussed in this paper.

Solution	Pros	Cons
Tracking network data	<ul style="list-style-type: none"> • Can detect unknown bots • Encryption isn't a problem 	<ul style="list-style-type: none"> • Tradeoff between coverage and performance
Analyzing network data	<ul style="list-style-type: none"> • Can block an infection attempt 	<ul style="list-style-type: none"> • Needs frequent signature updating • Encryption is a problem
Anti-virus	<ul style="list-style-type: none"> • Blocks known bots quite efficiently • Can block an infection attempt 	<ul style="list-style-type: none"> • Needs frequent signature updating • Variants can evade the detection
Behavior blocking	<ul style="list-style-type: none"> • Can detect unknown bots 	<ul style="list-style-type: none"> • Can make many false alarms • Action of the user is needed

Table 1: Summary of different approaches to detect peer-to-peer bots on the host

4 Conclusion

Botnets are among the most important threats to the security of the Internet, as launching points for many different attacks. They have started to employ peer-to-peer networking to overcome traditional IRC based command and control channel weaknesses, such as shutting down the IRC servers practically disables the whole botnet.

The usage of peer-to-peer command and control channel, and encryption makes it harder to detect a bot from network data. The solution is host based methods. These are tracking what is done with the network data on the host, and the traditional methods of antivirus and behavior blocking.

The antivirus and behavior blocking are widely used methods. The BotSwat approach is a promising addition to them to fight the peer-to-peer based botnets.

5 Future Work

The methods discussed in this paper could be tested on some peer-to-peer based bots currently running in the wild.

References

- [1] D. Cullen. Raa drops the dead edonkey, Sep 13th 2006. http://www.theregister.co.uk/2006/09/13/edonkey_settles_copyright_suit/.
- [2] F. Freiling, T. Holz, and G. Wicherski. Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks. *Computer Security-ESORICS 2005: 10th European Symposium on Research in Computer Security*, 2005.
- [3] J. B. Grizzard, V. Sharma, C. Nunnery, B. B. Kang, and D. Dagon. Peer-to-peer botnets: Overview and case study. In *HotBots '07 conference*, 2007.
- [4] M. Jonkman. Encrypted storm traffic, 2007. <http://www.bleedingthreats.net/index.php/2007/10/15/encrypted-storm-traffic/>.
- [5] K. Kasslin, M. Ståhlberg, S. Larvala, and A. Tikkanen. Hide 'n Seek revisited-Full Stealth is Back. *Virus Bulletin Conference October*, 2005.
- [6] J. Kuo. Storm drain. *Microsoft Technet Blogs*, Sep 20th 2007. <http://blogs.technet.com/antimalware/archive/2007/09/20/storm-drain.aspx>.
- [7] R. McMillan. Storm worm now just a squall. *Network World*, Oct 21st 2007. <http://www.networkworld.com/news/2007/102107-storm-worm-now-just-a.html>.
- [8] E. Messmer. Behaviour blocking repels new viruses. *Network World Fusion News*, Jan, 2002.
- [9] Microsoft Technet. *Peer Name Resolution Protocol*, 2006. <http://technet.microsoft.com/en-us/library/bb726971.aspx>.
- [10] M. Ripeanu and I. Foster. Mapping the Gnutella Network: Macroscopic Properties of Large-Scale Peer-to-Peer Systems. *First International Workshop on Peer-to-Peer Systems (IPTPS)*, 68, 2002.
- [11] M. Russinovich and B. Cogswell. Windows NT System-Call Hooking. *Dr. Dobbs's Journal*, 22(1):42-46, 1997.
- [12] B. Schneier. How bot those nets? *Wired Magazine*, July 27, 2006. <http://www.schneier.com/essay-120.html>.
- [13] O. Sreebny. Legitimate uses of p2p, Oct 2007. http://staff.washington.edu/oren/weblog2/archives/2007/10/legitimate_uses.html.
- [14] J. Stewart. The changing storm, 2007. <http://www.secureworks.com/research/blog/index.php/2007/10/15/the-changing-storm>.
- [15] J. Stewart. Storm worm ddos attack. Technical report, SecureWorks, Inc., 2007. <http://www.secureworks.com/research/threats/storm-worm>.
- [16] E. Stinson and J. C. Mitchell. Characterizing bots' remote control behavior. In *Lecture Notes in Computer Science*, volume 4579. Springer Berlin / Heidelberg, 2007.
- [17] Mininova breaks 3 billion downloads barrier, Nov 2007. <http://torrentfreak.com/mininova-3-million-downloads-071101/>.

- [18] P. Wang, S. Sparks, and C. C. Zou. An advanced hybrid peer-to-peer botnet. In *HotBots '07 conference*, Apr 2007.