Botnets: A Crash Course

Sason Baghdadi
sasonbaghdadi@csus.edu

Abstract. Botnets using client-server network models are incredibly powerful, easy to create and manage, and rampant in the wild. They use clever techniques to keep the botnet master(s) anonymous and difficult to trace. A demonstration using an IRC server as a source of communication for the infected machines shows how powerful a simple botnet can be.

Keywords: Botnet, IRC, Proof of Concept, Python, GameOver ZeuS

1 What is a botnet?

1.1 Definition

The word “botnet” is a combination of the words “robot” and “network” and is typically associated with negative or malicious connotation. A botnet is a network of infected machines, controlled by one or many hackers through command and control software (C&C), that work together to accomplish the goals of the botnet master(s).

Machines are infected and join a botnet when they execute malicious software by: visiting a website and executing a drive-by download, exploiting web browser vulnerabilities, or by tricking the user into running a Trojan horse program such as an email attachment.

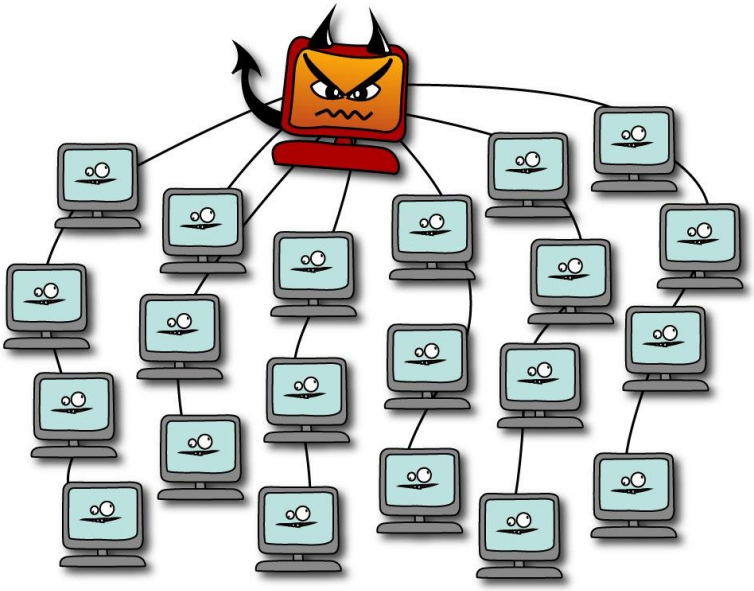

Fig 1. Client-Server Model

1.2 Client-Server Model

The most traditional and frequent model of botnets used the client-server model (also referred to as master-slave model) in which the client (infected machines) would receive commands from, and only communicate with, the server (botnet master). The botnet master sends command to the server, the server relays the commands to the clients, and the clients execute the commands and report their results back to the botnet master.

These botnets communicate through internet relay chat (IRC) networks, DNS protocols, or websites. In the case of IRC botnets, such as the demo PoC below, the infected clients connect to an IRC server and join a channel designed as a C&C for the botnet master. The botnet master sends commands to the IRC channel and the clients in the channel execute the commands and report their results to the master.

Botnet sizes can vary from a few hundred infected systems up to over tens of millions of infected systems, such as the BredoLab botnet that had over 30 million clients and infected an estimated 3 million PCs per month [1].

1.3 Peer-to-Peer Model

IRC-based botnets are easy to detect, which is the reasoning behind the development of botnets modeled after peer-to-peer networks. Peer-to-peer, such as GameOver ZeuS seen in section 4.2, use digital signatures so that only someone with access to the private key can control the botnet [2].

Each client in the botnet is communicating commands to other clients within the botnet, as opposed to a centralized server server like the client-server botnets. Newer botnets are typically seen using the peer-to-peer model because it makes it harder to find where the botnet commands are coming from. The command distribution also prevents a single point of failure, which is a big issue for client-server botnets.

2 What is a botnet capable of?

2.1 Spam

Infected machines are used to send email messages to millions of people disguised as messages from real people, but are really either advertising, annoying, or malicious.

The malicious email messages usually contain one, or many, of the following: file attachments such as .pdf/.zip/.doc with embedded malware, website URLs that automatically download and execute malware, or a Nigerian prince trying to steal banking information by offering a few million dollars in exchange. The malware involved in the malicious emails is one technique used to unknowingly infect a user's system and join a botnet.

2.2 Distributed Denial-of-Service Attacks

The vast majority of botnets are capable of executing distributed denial-of-service attacks (DDoS) in which the botnet master uses the clients' systems to submit as many requests as possible to a server, thus overloading it and preventing it from processing legitimate requests.

A recent example of a DDoS attack using a botnet is the Mirai botnet attack against Dyn DNS servers in 2016 which disrupted an entire day of service for: Amazon, CNN, GitHub, Netflix, Twitter, Visa, and many other high-profile companies [3].

2.3 Financial Gains

Financial incentives are what drive people to create the best product they can. That methodology also applies to botnet creators. Botnet creators make financial gains through: ransomware, providing other hackers access to clients within the botnet, bank fraud, click fraud, mining cryptocurrency, and stealing intellectual property.

Bank fraud and renting out the botnet to other hackers are the two highest-paying methods in which botnet creators make money. For example, the GameOver ZeuS botnet spread using a banking trojan which resulted in an estimated global loss of over $100 million [4].

3 How do botnets work?

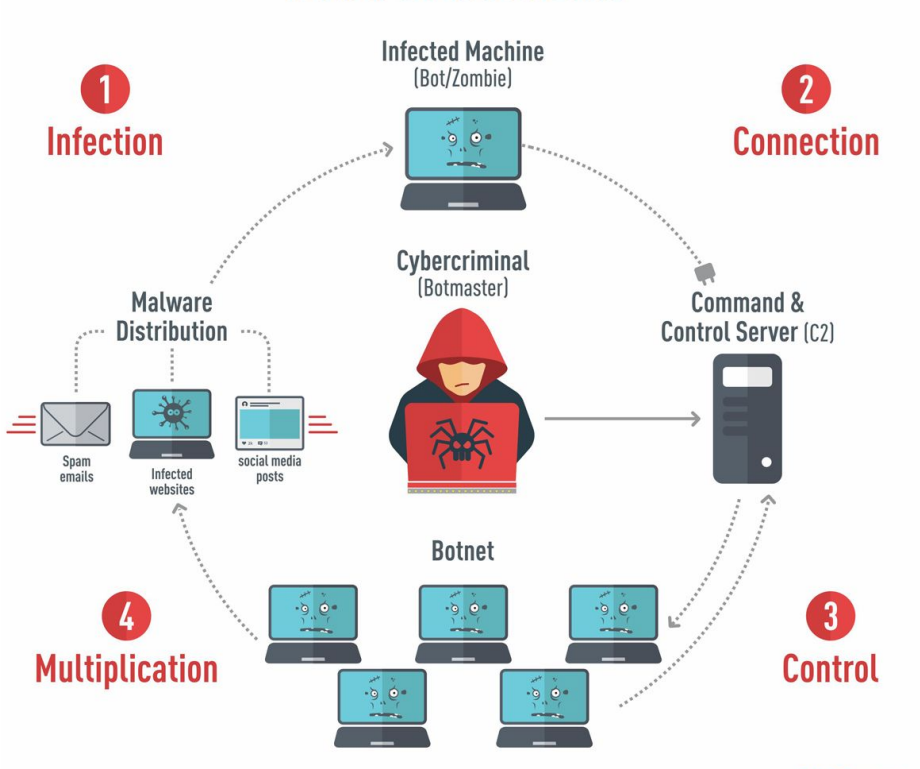

Fig 2. How botnets work: Infection, Connection, Multiplication, Control

3.1 Infection

Cyber criminals spread malware through spam emails with malicious attachments, drive-by downloads, malicious USB drives, and Trojans. A user will open an email attachment, or visit a malicious website and unknowingly download the malware.

3.2 Connection

The malware automatically sends a request to join a botnet. The request is sent in whatever medium is being used to communicate with the C&C server, such as IRC, email, pastebin, DNS, etc. Communication between the client and botnet master is typically heavily encrypted or smuggled inside of regular traffic.

After sending the request, the C&C server will request for authentication from the client to verify the system is in fact infected with the malware. Once verified with the C&C server, the user's computer joins the botnet and allows the botnet master access to the user's computer.

3.3 Control

After establishing a connection from the C&C to the client, the botnet master can now: issue remote commands, upload data, and download data from the user's machine.

Issuing Remote Commands. Issuing remote commands to the client allows the botnet master to spam emails via the client in order to infect more machines, broadcast commands for a targeted DDoS attack, and rent access of the client to other hackers (which will allow other hackers to upload their own malware).

Upload Data. Uploading data to the client allows the botnet master to send more malicious programs to the infected machines, such as keyloggers or remote administration tools (RATs), or send software updates for the malware.

Download Data. Downloading data from the client allows the botnet master to steal sensitive or confidential files from the client.

3.4 Multiplication

The botnet multiplies through many unique techniques created by, or made available to, the botnet master. Propagation techniques include spam emails with malicious attachments, searching through the machine's local network for vulnerable machines, or searching through the public network for vulnerable machines. The most used propagation technique is spam email with malicious attachments as it's extremely simple for a botnet master to hide malware within a .pdf/.zip/.doc file.

4 Real world examples

4.1 PyCryptoMiner

PyCryptoMiner is a crypto-miner botnet targeting Linux system that is spreading over the SSH protocol [5]. The botnet is based on the Python scripting language, leverages Pastebin.com as a backup C&C, and uses the infected machines to mine the cryptocurrency Monero. In mid-December, the botnet creator pushed a software update to the botnet that allows new functionality for hunting and exploiting vulnerable JBoss servers associated with CVE-2017-12149 [6].

The botnet creator is associated with more than 36,000 domains, some of which are known for scams, gambling, and adult services since 2012. As of late 2017, the botnet has made approximately $46,000 mining Monero. As of December 2018, PyCryptoMiner botnet has infected 1,000+ Linux systems and is still running rampant in the wild.

4.2 GameOver ZeuS

The GameOver ZeuS (GOZ) botnet is a brilliant evolution of the ZeuS Trojan, first identified in 2007 and retired in 2010 [7], designed in 2011 to steal banking credentials from users and deploy CryptoLocker ransomware. GOZ used an encrypted

peer-to-peer network model to communicate between the clients and C&C server while predominantly spreading through spam email or phishing messages.

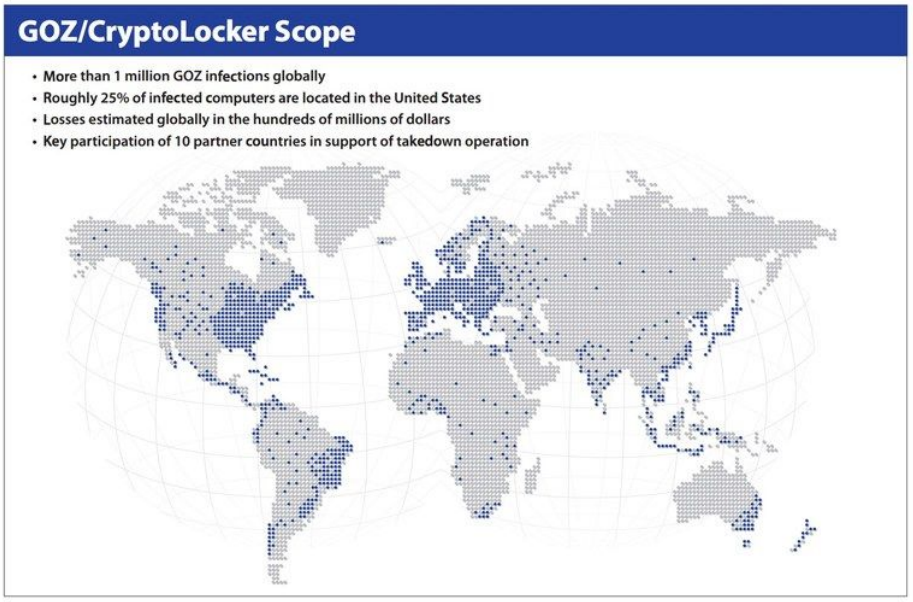

Fig 3. GameOver ZeuS scope [8]

Over the span of 3 years, from 2011 to 2014, GOZ infected an estimated 3.6 million computers with estimated global losses in the hundreds of millions of dollars [9]. Following 2 unsuccessful attempts to take down the botnet in 2011 and 2012, the FBI, alongside 10 partner countries and countless security agencies, finally took down the botnet in June 2014 [8].

The Russian-based botnet creator Evgeniy Mikhailovich Bogachev, also known as "Slavik", has not been found. The FBI has since put a $3 million bounty on his head, the highest rewards the US has ever posted for a cybercriminal [10].

5 Detecting botnet infections

5.1 Monitoring network traffic

Free tools, such as Wireshark, tcpdump, or a router's built-in intrusion detection system (IDS), can be used to monitor network traffic for suspicious websites or IP addresses. Suspicious IP addresses can be cross-checked with SonicWall's botnet IP lookup tool [11] in order to verify whether or not the network has been compromised.

5.2 Keeping an eye out suspicious processes

It's important to be able to notice and identify suspicious processes running in the background of a computer. For example, if a computer predominantly uses Firefox as

its web browser, and there's a process named 'iexplorer.exe' or 'chrome.exe' running in the background, it's likely the case that's the system is infected with malware.

5.3 **Noticing strange system hiccups/glitches**

Strange system hiccups, such as a computer slowing down, high CPU usage, or a mouse moving on its own and clicking on buttons, are signs of a computer being potentially infected with malware. It's important to be able to notice and identify such symptoms.

6 Preventing botnet infections

6.1 Preventing botnet infections

- Use common sense
- Update operating systems, software, firmware, etc. as soon as possible
- Don't click on untrusted or suspicious links
- Don't fall for spam emails
- Close all unused ports from the internet
    - If a port is required to be open for some service, ensure the service is updated and secured with authentication
- Change default passwords of anything and everything in sight
- Download an antivirus and antimalware software and actually use it
    - Recommended free software: Malwarebytes Anti-Malware and AVG

7 Proof of Concept Demo - IRC Botnet

7.1 Setting up the IRC Server

The botnet server is ran on Kali Linux (version: Linux Kali 4.18.0-kali2-amd64 #1 SMP Debian 4.18.10-2kali1 (2018-10-09) x86_64 GNU/Linux). The irc server used is inspircd and ran by executing 'service inspircd start'. To keep the demonstration simple and concise, default settings of the IRC server were left alone. The only changes made to the IRC configuration are seen below in figure 4.

Of course, it's important to note that real-world botnets are not this simple. Communication between the client and C&C is typically encrypted and stealthily transported so as to not raise suspicions or alerts within the network. In this demo, however, the traffic is not encrypted in any shape or form, so the client could easily detect that their system is part of a botnet by viewing their network traffic logs and seeing commands being sent and executed by someone named 'Master' in an unknown IRC server.

```
<server name="irc.botnet.local"
        description="Local IRC Botnet Server"
        network="Localnet">

<admin name="Botnet Master"
       nick="Master"
       email="root@localhost">

<bind address="127.0.0.1" port="6667" type="clients">
```

Fig 4. inspircd default configuration settings

7.2 Botnet source code

The botnet source code was written in Python 3.6.7. The source code can be found at: https://github.com/nosas. The major functions in the source code, along with their purpose in the program, are as follows:

- *SLAVENICK:* A unique identifier provided to the bot upon joining the botnet.
- *join_server*: Opens a socket to, and authenticates with, the IRC server.
- *join_channel*: Joins the designated channel (C&C server).
- *send_message*: Sends a message to the C&C or directly to the botnet master within the IRC server.
- *receive_message*: Receives a command from the botnet master.
- *execute_command*: Executes a shell command provided by the botnet master.
- *open_reverse_shell*: Open a reverse shell to a designated IP address, allowing whoever is listening from the designated IP address to execute commands on the infected machine.
- *retrieve_system_info*: Downloads linEnum.sh (a system information parser and vulnerability scanner) to /tmp/, executes the script, temporarily uploads the output to pastebin.com, and sends the pastebin URL to the botnet master.
- *upload_to_pastebin*: Uploads a file to pastebin.com.
- *run*: The main program that continuously listens for commands from the C&C and reacts accordingly.

The botnet master is able to invoke commands on the infected machine by sending the following messages into the designated IRC channel:

- *!exec*: Executes a shell command on the target. If sent to the channel, the command is broadcasted and executed to all clients within the channel. If sent as a private message to a client, only that client will execute the command.
- *!execo*: Executes a command with the same guidelines as above, except it sends output back to the botnet master.
- *!kill <SLAVENICK>*: Severs the client's connection from the botnet.

- *!shell <SLAVENICK> <DESTIP> <DESTPORT>*: Opens a reverse shell to the destination IP and port from the client's system. A listening port must be open on the destination address for the reverse shell to work.
- *!system <SLAVENICK>*: Downloads linEnum.sh to the client and waits for a pastebin URL to be sent back to the botnet master.
- *!help*: Displays the usage of bot commands.

7.3 Joining the botnet with the infected machine

Upon downloading and opening the malicious file, the user's system will automatically execute the following command in order to run the malware: "python3 irc_botnet.py". It will then connect to the IRC server, authenticate with the botnet, join the designated C&C channel, and send "Hello World" to the channel to indicate it's ready to receive commands. It's important to note that this is all running in the background of the victim's computer without their knowledge.

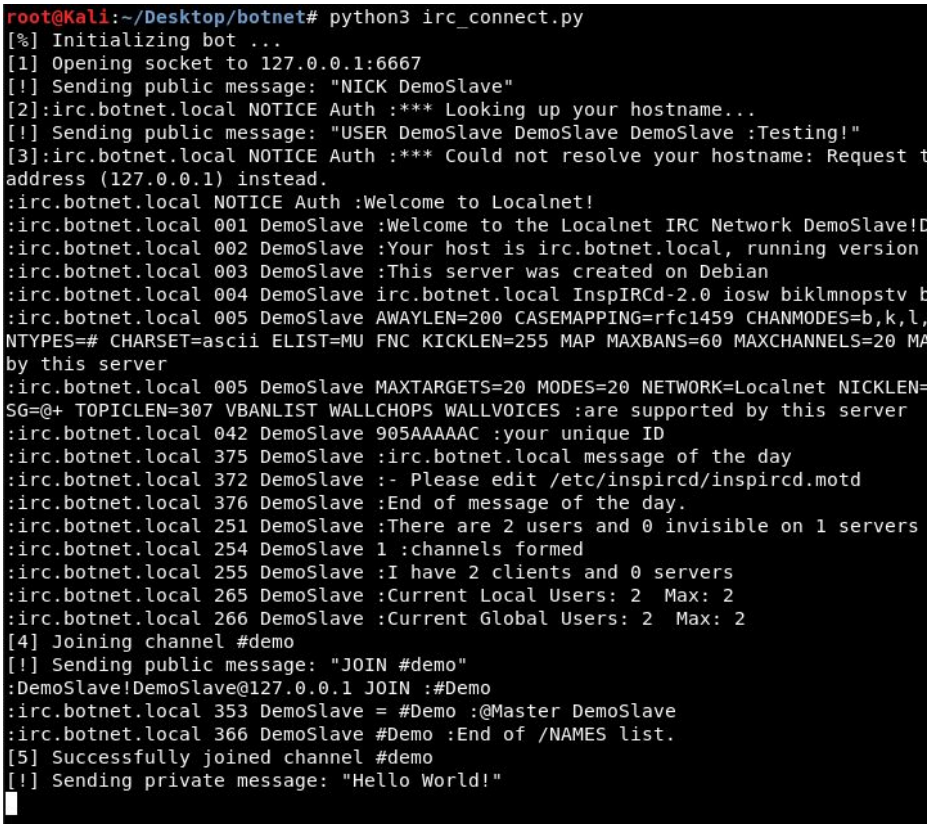

Fig 5. Client connecting to the IRC server, joining the channel, and sending "Hello World"

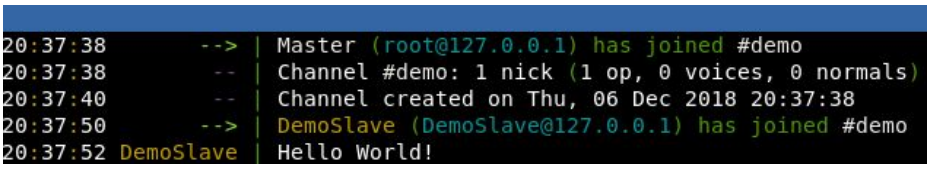

Fig 6. The client's message as seen from the botnet master's perspective

7.4 Executing commands on the infected machine

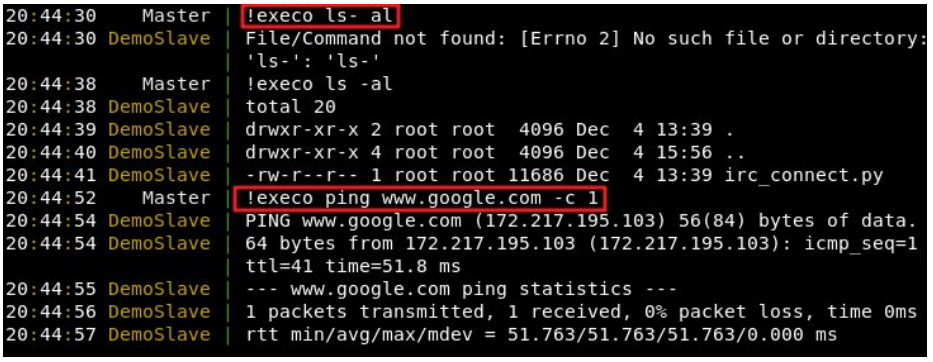

Fig 7. Executing '*ls -al*' and '*ping www.google.com -c 1*' on a client's system

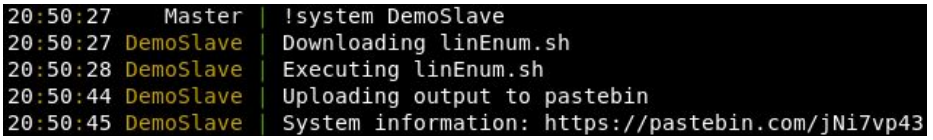

Fig 8. Requesting system information from the client, and the client returning a URL to the output on pastebin.com

The final command to be demonstrated is the reverse shell. First, the botnet master needs to open a socket that listens for any incoming connection by using the command '*nc -l -p PORT*'. After opening a socket that listens for incoming connections, the botnet master will send the following command to a client, '*!shell SLAVENAME DESTIP DESTPORT*' and the reverse shell will open.

Fig 9. Opening a reverse shell to the designated IP address and port

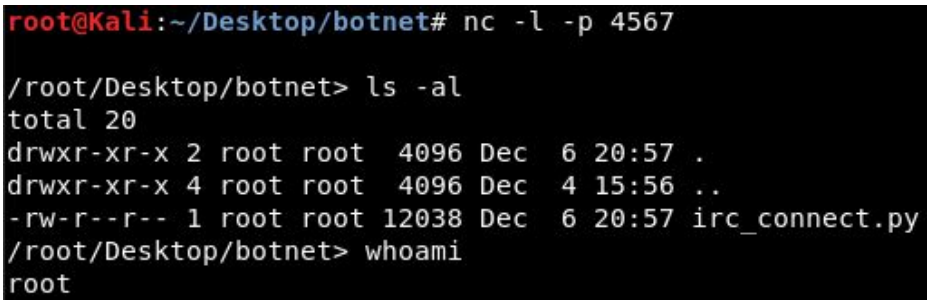

Fig 10. Reverse shell successfully opened to the client

Fig 11. Severing the client's connection from the botnet

References

1. Krebs, B: Bredolab Mastermind Was Key Spamit.com Affiliate, 30 October 2010.
    a. https://krebsonsecurity.com/2010/10/bredolab-mastermind-was-key-spamit-com-affiliate/
2. Heron, S: Botnet command and control techniques, April 2007.
    a. https://www.sciencedirect.com/science/article/pii/S1353485807700454
3. Krebs, B.: Hacked Cameras, DVRs Powered Today's Massive Internet Outage, 21 October 2016.
    a. https://krebsonsecurity.com/2016/10/hacked-cameras-dvrs-powered-todays-massive-internet-outage/
4. Sandee, M.: GameOver ZeuS: Backgrounds on the Badguys and the Backends, 2015.
    a. https://www.blackhat.com/docs/us-15/materials/us-15-Peterson-GameOver-Zeus-Badguys-And-Backends-wp.pdf
5. Brailsford, A., Segal, L., Zavodchik, M.: New Python-Based Crypto-Miner Botnet Flying Under the Radar, 3 January 2018.
    a. https://www.f5.com/labs/articles/threat-intelligence/new-python-based-crypto-miner-botnet-flying-under-the-radar?sf178360556
6. NIST National Vulnerability Database: CVE-2017-12149, 4 October 2017.
    a. https://nvd.nist.gov/vuln/detail/CVE-2017-12149
7. Finkle, J.: Hackers steal U.S. government, corporate data from PCs, 16 July 2007.
    a. https://www.reuters.com/article/us-internet-attack/hackers-steal-u-s-government-corporate-data-from-pcs-idUSN1638118020070717
8. FBI: GameOver Zeus Botnet Disrupted, 2 June 2014
    a. https://www.fbi.gov/news/stories/gameover-zeus-botnet-disrupted
9. Lawrence, D.: The Hunt for the Financial Industry's Most-Wanted Hacker, 18 June 2015
    a. https://www.bloomberg.com/news/features/2015-06-18/the-hunt-for-the-financial-industry-s-most-wanted-hacker
10. Graff, G.: Inside the Hunt for Russia's Most Notorious Hacker, 21 March 2017
    a. https://www.wired.com/2017/03/russian-hacker-spy-botnet/
11. SonicWall: Botnet IP Status Lookup, 2017
    a. http://botnet.global.sonicwall.com/view