Human vs. Botnet Attacks

Noah Burkhardt, Lillian Henry, Ethan Waldner, Noah Zbozny

Group 2F

HACS200 / Fall 2018

Executive Summary

The purpose of our project was to, in a comprehensive way, statistically quantify the differences in behaviors exhibited that characterize automated (botnet) versus real-time attacks. Therefore, we focused on two main identifying aspects that could be used to draw conclusions about the distinctions between a botnet attack and a human attacker, and our research question became: How do botnets and real-time cyber intrusions differ in their session duration as well as the commands typed?

We first sought to determine whether there was notable and statistically significant contrast in the actions taken when an attacker entered the honeypot. Consequently, our null hypothesis was that there is no difference between a botnet and human attack with respect to behavior and duration. Next, since the first facet of our project was to study the duration of the intrusion, we predicted that the botnets would more efficiently accomplish their objectives and therefore our first hypothesis was that the session duration of a botnet is statistically significantly different than that of a human intruder. And finally, complementary to the idea that botnets may be faster than humans, we predicted that botnets likely also operate in a more systematic approach and take different steps to achieve their objectives; thus, our second hypothesis was that the activity/objectives of botnets versus humans is significantly different.

Our statistical results were interesting in that our use of chi-squared test for independence in addressing our first hypothesis resulted in a rejection of our null hypothesis and the determination that the types of commands used is in fact dependent on the type of attack. For the second hypothesis, we used an unpaired sample t-test, which resulted in a high p-value and thus we were unable to reject the null hypothesis for our session duration hypothesis.

Background Research

Initially, we reviewed scholarly sources in order to ensure that our honeypot setup and question definition would produce relevant data that we could appropriately analyze and use to produce a useful conclusion. We consulted the research paper *Application of Routine Activity Theory to Cyber Intrusion Location and Time* in clarifying our concepts of human and botnet attacks and refine our definition of botnets, which is central to how we categorized our attack data. The researchers' quantified limits in identifying which sessions and connections were attributed to botnets inspired our own constraints, including our utilization of typing speed and use of the backspace key in order to have multiple methodologies to attain a high accuracy of differentiating between humans and botnets. The paper also helped us cement our research question as we needed to make sure that live human attacks as well as botnet attacks would be common enough to meet frequency requirements in order to apply our desired statistical tests. In the results of this paper, around 33% of all intrusions were from botnets, with the percentage varying greatly based on source region but nevertheless providing a substantial enough sample size to make insightful use of statistics.

Once we were able to properly frame our research with consistent definitions and appropriate conditions to break down our data, we sought additional research to intuit how we might expect the botnets to differ in behavior from the human attackers. In *Botnet Detection by Monitoring Group Activities in DNS Traffic*, researchers from Korea University explored how understanding of botnet objectives and motivations could prove beneficial with detecting when bots connect to their server or migrate to others in the face of fast-evolving botnet technologies. The paper's 'introduction,' 'features of botnet DNS,' 'discussion' and 'conclusion' sections were

particularly valuable in revealing that detection of botnets is made easier by algorithms that take into account the fact that much of recent malicious botnet activity seeks to reap financial benefits from sending spam, stealing personal information, launching DDoS attacks, and more. This is pertinent to our research question as we hoped to identify such trends in the data that would show that botnet activity might tend to be more systematic and efficient compared to the activity displayed during a human intrusion. With botnets being controlled by pre-written programs, we predicted that we would see indications that these programs would not mimic the typical actions a human would take, and instead execute their primary objective in pursuit of maximum exploitation. Further, the systematic approach that we predicted the botnets would adhere to has to do with the speed and duration of intrusion; with pre-planned scripts being executed, the botnets would quickly execute the necessary commands to accomplish the objectives they have, whereas in contrast a human attacker would take more time to attack, whether due to slower typing speed or evolution of the person's objectives throughout the attack.

Another paper that we consulted throughout our project was *Lessons learned from the deployment of a high-interaction honeypot*, which reflected upon a 6-month controlled experiment run with a high interaction honeypot compared to findings established by a worldwide distributed system of low-interaction honeypots. With this resource we hoped to learn from other researchers' experiences and preemptively correct any mistakes we might make in the process of gathering and analyzing data so that we could produce practical and useful statistics. The French authors provided a comprehensive explanation of the stages of deployment and

details about how they were able to characterize the intrusions that they saw with respect to the attackers' activities and skills. We referred to this paper while drafting our initial design proposal as well as when refining our question and strategy as it explained the context for how different experiments are designed to utilize a low-interaction versus a high-interaction honeypot. Finally, we used this paper towards the end as we wrapped up our deployment and began drawing conclusions, as the researchers broke down specifically how they measured the parameters of interest in their experiment and how the data was used for their tests and figures. Overall it was an interesting read and valuable supporting resource that prepared us for the challenges of pursuing our research interest with a high-interaction honeypot.

Experiment Design Changes

The experiment design laid out in the project proposal was to have the generic honeypot system of four honeypots with the Man-In-The-Middle (MITM) architecture set up on each machine. The most impactful design change that needed to be made to the project was to change the categories of commands that attackers entered in the honeypots. This retroactive shift in the command categories arose after perusing our session data and coming to the conclusion that we would be able to draw stronger statistical conclusions while still maintaining relevance to our research question by combining certain areas that corresponded in perceived attacker 'objective.' The original command categories consisted of "exploratory," "downloading/file transfer," "editing/compression," "searching/regex," "system management," and "scripting." As mentioned, during our first analysis of the data, we realized that there were commands being

entered that did not entirely conform to any of these initial categories, so we added a "miscellaneous" category in order to maintain the integrity and definitions that would correspond to our hypothesized behavioral differences. Additionally, once we collected more data and performed more thorough analysis of the data, we found that human actor input was actually entirely absent in the categories "downloading/file transfer," "editing/compression," "searching/regex," and "scripting," while botnets never executed "downloading/file transfer" or "editing/compression" commands. Our solution to this problem was to combine some of the categories into more general categories of commands so that we could perform the chi-squared test analysis on our data with the required threshold of data points in each category. We combined "exploratory" with "searching/regex," "downloading/file transfer" with "editing/compression" and "system management," and "scripting" with "miscellaneous." This resulted in a more easily understandable distribution and remained true to our aim to identify whether human and botnet attack sessions had fundamentally different intentions. Other minor changes that occurred throughout the process of reevaluating our design proposal and receiving feedback from instructors included a shift in our recycling policy from a relative timeframe to event-based recycling, which made the recycling a better fit that would maximize our opportunity for data collection.

Experiment Design

The experiment is set up to have 4 honeypots, each using the default firewall rules provided by ACES and each running the Man-In-The-Middle (MITM) software provided by ACES to log timestamps and keystrokes input into the honeypot. Being able to log time

difference between keystrokes allows for analysis of typing speed of attackers, which is how we determined which actors were humans and which actors were botnets. The honeypots were all set up to be regularly pinged by Uptime Robot in order to monitor their health. In addition, we used Uptime Robot to ensure that the SSH port was still open and accepting connections. As a final health measure, we set up a health log script that allowed us to view the health of the honeypot at 3 hour intervals. This health log data included total disk space used by the honeypot, total RAM usage per honeypot, and total CPU usage per honeypot. For each individual honeypot, the host was regularly collecting attacker input through the MITM software, and at various intervals the team would download the data collected by the host onto a local machine for both storage and analysis. Our data analysis script was based around the use of regular expressions. We used regular expressions to parse the commands out of the MITM-generated files. We used the same method with a different regular expression to parse out the timestamp of that entry in the logs. Using this data, we determined the words-per-minute being typed by the attacker by taking the difference in the timestamps. As mentioned previously, we used this metric to differentiate between botnets and human attackers. The cutoff for humans and botnets was set at 300 words-per-minute; specifically, attackers with a typing speed above 300 words-per-minute were categorized as botnets, and the attackers with a typing speed below 300 words-per-minute were labeled human attackers.

Each container was set up in a standard fashion, with the default SSH configuration and firewall rules. Recycling the honeypots occurred whenever an attacker exited the machine. The recycling process involved resetting the states of the MITM instances. We simply wrote a small

script to accomplish this. Additionally, we created backups for each honeypot were created when the containers were each created, and upon the honeypot being recycled it was restored to the state of the backup (the clean state). This recycling process occurred whenever an attacker ended their session (logged out of the container) or at every hour. Finally, we created a setup script, which contained all of the firewall rules and routing table entries for the whole system. This script ran on startup to ensure that we always started with the same state upon reboot.

For clarity, here are some definitions we used for this experiment.

Intrusion: An attacker successfully entering the honeypot using SSH.

Botnet: We define botnet as an attacker whose typing speed exceeded 300 words-per-minute. We chose this number based on research we completed on typing speed throughout the course of the project.

Human attacker: We define a human attacker as the complement to the botnets. That is, we will monitor their typing speed, and categorize them as a human attacker if their typing speed does not exceed 300 words-per-minute. As an added measure, due to the fact that humans are error-prone, we will look for use of the backspace key as well as longer pauses between keys, which both are indicative of an attacker typing in real time.

Session duration: Session duration is the total time the attacker spent on the compromised honeypot. It will be calculated as the difference between the initial and final timestamps of the attacker session logs.

Commands: Categories will include those used to download or execute scripts, exploratory commands, and attempted alteration of structure and security; these will be enumerated and expanded upon later in the 'Data Collection' section.

Data Collection

Most of the data collection work was performed automatically, via the MITM software. The MITM logged the session duration for each session, the container that the attacker logged in to, the username and password used by the attacker, the IP address of the attacker, and every keystroke entered by the attacker along with the timestamp for each keystroke. The data collected that was most relevant to this project was the session duration, keystrokes entered by the attackers, and the timestamp for each keystroke. All of the data was collected by the MITM software and was stored on the host machine. It was stored in two forms: first, as a single log file per honeypot (meaning four in total) which contained every entry since the start of the experiment, and second as compressed files for each individual session. Whenever a member of the team analyzed the data, it was pulled down to their local machine and kept there for additional storage as well. By the end of the experiment, there ended up being 994 sessions collected in total over all four honeypots. Of those 994 sessions, 174 sessions had at least one command entered in them (including "exit"). Of those 174 sessions, 102 of them had at least one command entered other than "exit". The 102 data points that had at least one command entered other than "exit" are the ones that are relevant to our research question, so the rest of the data points were eliminated from our statistical analysis. Processing of the data was done using several python scripts written to parse and analyze the data. First, a team member would use

SFTP to connect to the host machine and pull all of the attacker sessions data down from the host to a local machine. The files were saved in the form of a .gzip file, so first we wrote a script to unzip and decompress all of the files. This script would also automatically delete some of the empty sessions which were too small to be unzipped (typically around 10 bytes). These files were likely created due to some malfunction by the MITM software. The remaining files would then be analyzed by a different script to log the keystrokes input by the attacker and their timestamps. This script would also convert the session durations from "YYYY-MM-DD HH:MM:SS.s" format, which was the output format of the MITM software, into UNIX time format, which displays it as milliseconds since the UNIX epoch (January 1, 1970). This step aided in our parsing process by allowing the timestamps to be easily subtracted so that we might find the time between keystrokes. The next step was for another script to take the output from the aforementioned script, determine the words-per-minute for each session, and then use that words-per-minute analysis to differentiate between botnet and human, and output that (along with the commands entered in that session and the session duration). The way that the script differentiated between human and botnet was based on word count and the number of backspaces that were logged by the MITM. The words-per-minute cutoff for determining if an actor was a botnet or a human was 300 words-per-minute (lower indicated human, higher indicated botnet). Additionally, if the session contained any backspaces, it was determined to be a human, as it was assumed that botnets would not make mistakes. This assumption later proved to be true upon human analysis of the data, as all of the sessions that had backspaces had words-per-minute counts lower than the threshold of 300. Then, a member of the team looked at the output of this script and determined the number of commands entered per category for each

session. The final result was a Comma-Separated Values (CSV) file that contained an indicator of whether the attacker was a human or botnet (denoted as ‘h’ or ‘b’), the session duration, and the counts of each category of command input in that session. This CSV data could then easily be transferred over to the Google Sheets form that the group used to store the data and perform statistical analysis.

Data Analysis

We had collected two types of data to comprehensively examine the difference between human and botnet attacks. The first type, commands used, was organized categorically so that Chi-Squared test for independence. The second type, attack duration, was simply divided by attack and tested using an unpaired sample T-Test.

We divided the commands used into three categories to analyze. These categories consisted of exploratory commands, system management commands, and miscellaneous commands. Exploratory commands consisted of any commands used to move about the honeypot or search for specific parts of the honeypot. Changing directories or “grepping” data would be examples of commands that would go under this category. System management commands consisted of those commands that would leave an affect on the honeypot itself. This could include downloading files or making new directories. Any other command used could be placed in the miscellaneous category.

Our initial thoughts were that the commands used between humans and botnets would differ significantly. This is because we figured a human and botnet type attack would have

different goals, and thus use different commands. For this reason, we made our null hypothesis for our Chi-Squared test for independence:

The commands used between human and botnet attacks would not be significantly different.

The data we collected supplied 28 human attack sessions as well 74 botnet attack sessions, for a total of 102 attack sessions with relevant data. Within those sessions a total of 149 commands were executed, to make an average of 1.46 commands per session. After arranging organizing the commands executed by attacker type and command category, we were then able to calculate expected values for each attacker type and command category based on the observed data points. Further calculations left us with a Chi-Squared value of 15.48.

Due to the nature of this experiment, we had two degrees of freedom. By referencing a Chi-Squared Distribution table, we can see that with an assumed alpha value of 0.05, the critical Chi-Squared value is 5.99. This means that any Chi-Squared value over the critical value would be significantly unlikely to align with the expected values. Therefore, since our Chi-Squared value is greater than the critical value, we can reject our null hypothesis.

To compare the session duration of humans versus botnets, an unpaired sample T-Test would work best. This is considering the data is single quantities, and each sample was independent of each other. Using data from 28 human attack sessions and 74 botnet attack sessions, we were able to create a statistical test. Initially we believed that the duration of a session between a human and a botnet would be significantly different, due to the speed of a botnet compared to a human. For this reason, our null hypothesis for our unpaired sample T-Test was as follows:

There will be no significant difference between the duration of a session when comparing a human attack to a botnet attack.

Using the data from all attack sessions, we were able to calculate several helpful statistics, including mean, variance and our T-statistic value. Altogether the data garnered a T-Statistic of 0.0082. Along with this, the human attack sessions recorded an extremely high variance value, totaling out 49248.17, meaning our data points are spread out very far from the overall human session mean. Regardless, we continued with the unpaired sample T-Test to see how the results would fair.

With 100 degrees of freedom and an assumed alpha of 0.05, a T-Statistic distribution table will give us a T-Critical value of 1.984. This means if a T-Statistic value were higher than this 1.984, it would be statistically likely that there is a difference in the data. However, since our T-Statistic of 0.0082 is much lower the T-Critical value, we failed to reject our null hypothesis for the unpaired Sample T-Test.

However, after looking at the large variability of the human attack session duration, we decided to run a new unpaired sample T-Test, but this time removing any outliers from the original data. By calculating the interquartile range, we can create the bounds for outliers in the original human data points. After calculating an interquartile range of 23.1, and further calculated that the upper bound for outliers would be any session that lasted over 61.6 seconds.

Using these new bounds, we were able to take out five outliers in the human session's category. This brought our total number of valid human sessions to 23, but considerably lowered our variance value to 227.95. This was still significantly higher than the botnet session variance however, which was only 0.441.

After a recalculation for the T-Statistic using 95 degrees of freedom, our final value turned out to be 0.284. This was still much lower than the T-Critical of 1.984. Because of this, our revised data set still failed to reject the null hypothesis.

Conclusion

For our Chi-Squared test for independence, we were able to reject our null hypothesis. Based on this we can conclude that based on the data, commands used are dependent on the attack type (human or botnet).

There was a particular disparity between the System Management and Miscellaneous categories between the two groups. While human attacks were expected to perform much more System Management commands than what was observed, botnet attacks were expected to perform much less System Management commands than what was observed. Vice versa applies to the Miscellaneous command category.

Our team interprets these results as this could possibly mean there is a difference in motives between human attacks and botnet attacks. Since the commands used are dependent on each group (based on the data), you can align commands used with what each individual attack is trying to accomplish.

If we were able to continue this research there is a number of new features, we would implement to garner more detailed results. The main thing we would do is make more detailed command categories. The categories we used were fairly broad, and thus there was only three ways to differentiate the commands used between all attacks. If we were able to more closely analyze each individual commands used by this sample, we could create more accurate command categories in order to possibly receive a stronger result.

For our unpaired sample T-Test, we were unable to reject the null hypothesis. Even after eliminating outliers from the data, the null hypothesis was unable to be rejected. Because of these results, we are unable to conclude/interpret anything about this data. We believe this result is mainly in part due to the high variance in human attack session duration, while the botnet duration variance is extremely low.

If we had more time allotted for research, one major change our group could have made would be to make complex "honey" or data for our attackers to search through. This possibly would change the amount of time an attacker stayed on the system, as they might spend more time seeing what was in each honeypot. This could possibly create a significant difference in session duration and thus make us able to reject the null hypothesis.

Appendix A

Some of the patterns in the attacks were indeed surprising to the group. The variance in the session duration for human attackers, by the end of the experiment, was very large. In contrast, the variance and standard deviation for the session duration for botnets were surprisingly small. This dichotomy proved to be an interesting result of our experiment. Similarly, some of the data itself proved to be noteworthy. We discovered a large number of entries of one specific command "ip cloud print". After further research, we learned that this was a botnet determining if it was connected to a specific router brand, presumably because this router has some vulnerability that the botnet was trying to exploit. This was another interesting and unexpected result of our experiment.

One of the largest pitfalls that the group encountered was issues with monitoring the honeypots. A schedule was never established to determine who would monitor the honeypot and when, so

there were times where the honeypots went unmonitored and were unable to collect data for a period of time.

Appendix B

Appendix C

Parse Script:

```
try:
    path = sys.argv[1]
except:
    print('Usage: ./parse.py [path]')
    exit(1)

if path == None or path == 'parse.py':
    print('Usage: ./parse.py [path]')
    exit(1)

os.system('ls ' + path + '/* > ls.txt') #list all the files in the folder

f = open('ls.txt', 'r')
lines = f.read().splitlines()
f.close()

os.system('rm ls.txt') #clean up

#results_file = open('results.csv', 'w+')
results = ''

for l in lines:
    f = open(l, 'r')
    text = f.read()
    m = re.findall('(\d{4}\-\d{2}\-\d{2} \d{2}:\d{2}:\d{2}\.\d{3}): (.*)', text)
    for tup in m:
        for val in tup:
            results += '"'
            fixed_val = val
            fixed_val = fixed_val.replace('\\', '\\\\')
            fixed_val = fixed_val.replace('"', '""')
            results += fixed_val
            results += '",'
        results = results[0:-1]
        results += '\n'
    if len(m) != 0:
        results += '\n'

results_file = open('results.csv', 'w+')
results_file.write(results)
results_file.close()
```

Session Durations Script:

```
#! /usr/bin/python2

import re
from datetime import datetime

f = open('results.csv', 'r')
lines = f.read()
f.close()

lines_list = lines.split('\n\n')
session_durations = ''

fmt = '%Y-%m-%d %H:%M:%S.%f'
i=0
for l in lines_list:
    i+=1
    m = re.findall('(\d{4}\-\d{2}\-\d{2} \d{2}:\d{2}:\d{2}\.\d{3})', l)
    if (i == 37):
        print(l)
    if len(m) > 0:
        tdelta = datetime.strptime(m[len(m) - 1], fmt) - datetime.strptime(m[0], fmt)
        print('%d: %s' % (i, str(tdelta)))
        split_lines = l.split('\n')
        session_durations += '"'
        for val in split_lines:
            x = val.split(',')
            session_durations += x[1].strip('"')
        session_durations += '",'
        session_durations += str(tdelta)
        session_durations += '"\n'
print(session_durations)

f = open('session_durations.csv', 'w+')
f.write(session_durations)
f.close()
```

WPM Script:

```
#! /usr/bin/python2

import re

f = open('session_durations.csv', 'r')
text = f.read()
f.close()

lines = text.split('\n')
wpm = 'h or b,CPM,Session Duration,Number of DELs,Commands,Session Duration (secs)\n'

results = []

for line in lines:
    if len(line) > 0:
        results.append([0,0,0,'',0])

i = 0
for line in lines: #find backspaces
    if len(line) > 0:
        m = re.findall('(\[DEL\])', line.split(',')[0])
        results[i][2] = len(m)
        i += 1

i = 0
for line in lines: #fill with session duration and commands
    if len(line) > 0:
        results[i][1] = line.split(',')[1]
        results[i][3] = line.split(',')[0]
        i += 1

i = 0
for line in lines: #find characters per minute
    if len(line) > 0:
        m = re.findall('(\[ESC\]|\[Space\]|\[CR\]|\[DEL\]|\[LF\]|\[ETX\])', line.split(',')[0])
        char_count = 0
        for val in m:
            char_count += len(val)
        char_total = len(line.split(',')[0]) + len(m) - char_count #total chars + special chars like enter, del - len
        dur = re.findall('(\d*):(\d*):(\d*)\.(\d*)', results[i][1])
        duration = float(dur[0][3]) / (10 ** len(str(dur[0][3])))
        duration += float(dur[0][2])
        duration += float(dur[0][1]) * 60
        duration += float(dur[0][0]) * 3600
        results[i][0] = (char_total / duration) * 60
        results[i][4] = duration
        i += 1

for res in results:
    if (res[0] < 1200): #1200 characters per minute is about 300 words per minute and is 20 characters per second
        wpm += 'h,'
    else:
        wpm += 'b,'
    wpm += str(res[0]) + ','
    wpm += str(res[1]) + ','
    wpm += str(res[2]) + ','
    wpm += str(res[3]) + ','
    wpm += str(res[4]) + '\n'

f = open('wpm.csv', 'w+')
f.write(wpm)
f.close()
```

Works Cited/References

Alata, Éric, et al. “Lessons learned from the deployment of a high-interaction honeypot.” IEEE, IEEE Press, 21 Nov. 2007, ieeexplore.ieee.org/document/4020829.

Bock, Kevin, et al. “Application of Routine Activity Theory to Cyber Intrusion Location and Time.” IEEE, IEEE Press, 1 Dec. 2017, ieeexplore.ieee.org/document/8123566/.

Choi, Hyunsang, et al. “Botnet Detection by Monitoring Group Activities in DNS Traffic.” IEEE, IEEE Press, 21 Nov. 2007, ieeexplore.ieee.org/document/4385169/#full-text-section.