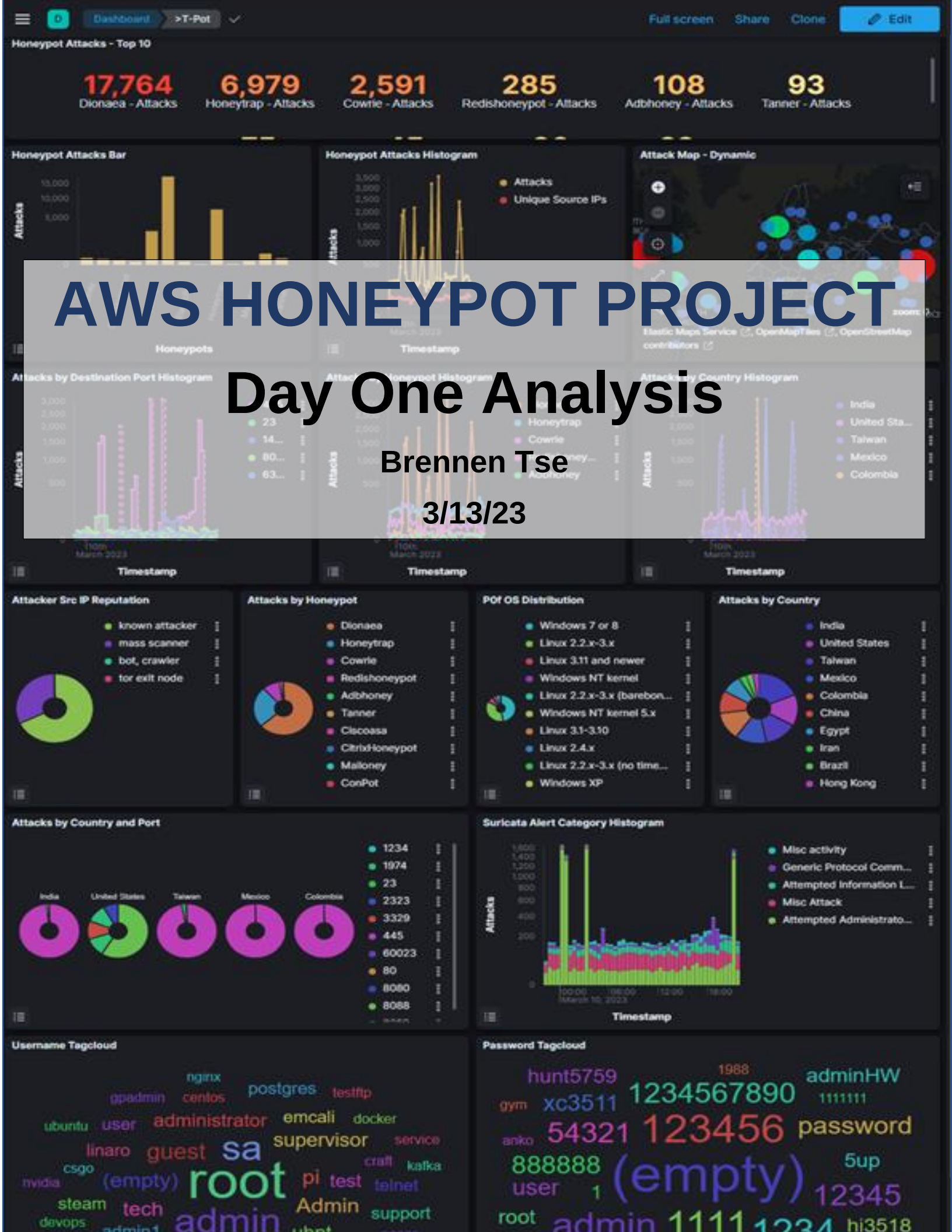


AWS HONEYPOT PROJECT

Day One Analysis

Brennen Tse

3/13/23

Day 1 Observations

Time: 3/9/23: 22:28 -> 3/10/23:22:28

Background Info: What are honeypots and why are they important in cybersecurity?

Honeypots are beneficial for cybersecurity because they provide a way to study the tactics and techniques used by attackers without risking actual systems or data. By analyzing the data collected by honeypots, security professionals can identify new threats, vulnerabilities, and attack patterns, which can be used to enhance security measures and develop more effective countermeasures. Additionally, honeypots can be used to detect attacks that traditional security measures might miss, such as zero-day attacks and advanced persistent threats.

T-Pot is an open-source honeypot platform that deploys multiple honeypots with various network services and protocols to attract attackers and record their activities. T-Pot's versatility in deployment extends beyond any restrictions of cloud platforms or virtual machines, as it can be conveniently installed on a wide range of Debian instances, including those in AWS, Azure, and other cloud platforms, thereby providing a flexible and scalable solution to honeypot deployment.

Lab Summary:

This report presents an initial assessment of a T-Pot honeypot's performance installed on a Debian AWS Instance, focusing on the first 24 hours of activity between March 9th, 10:00PM, and March 10th, 10:00PM. By analyzing trends and significant activity, the report aims to provide insights into predicting future cyber-attacks and determining the underlying factors contributing to the elevated volume of attacks observed on certain days of the week.

By also running a parallel T-Pot instance on the West Coast, my study aims to determine the potential impact of geographic location on the frequency and nature of cyber-attacks, providing insights into how network security can be tailored based on location-specific risk factors. The report's insights can aid in formulating comprehensive measures to enhance network security, including pinpointing systemic weak spots, identifying compromised usernames and passwords, and evaluating the effectiveness of existing security controls in thwarting brute force attacks.

This report will specifically outline what each honeypot does, analyze the overall T-Pot statistics and those of the top three attacked honeypots (Dionaea, Honeytrap, Cowrie) as well as determine outliers and their causes, highlight encountered issues and their resolutions, and provide predictions on the anticipated trends over the next week based on the data analyzed.

Table of Contents:

Overview

Defining the Top 10 Honeypot Types:

Dionaea is a low-interaction honeypot that emulates vulnerable Windows environments and services, capturing information on malware and attack payloads. It uses Python as a scripting language, libemu to detect shellcodes, and supports IPv6 and TLS, and logs include hash values of detected files that can be used for further intelligence gathering.

Honeytrap is designed to emulate various types of network services and protocols to attract and trap attackers. These emulated services include SSH, Telnet, FTP, HTTP, SMTP, and other common protocols. The Honeytrap module can be configured to listen on multiple ports and can be customized to mimic the behavior of real services as closely as possible.

Cowrie is a high-interaction SSH and Telnet honeypot designed to capture and record attacker activity on a simulated system. It provides a simulated shell environment that records all the attacker's actions, including login credentials, tools used, and techniques employed to gain unauthorized access.

Redishoneypot is a honeypot tool that emulates Redis servers to detect and track attackers targeting Redis databases. It captures information about attackers, their IP addresses, and commands issued during connection attempts.

ADB Honeypot is a security tool that emulates an ADB-enabled Android device to detect and log unauthorized access attempts.

Snare/Tanner is a low-interaction honeypot designed to emulate Windows-based systems and services, which captures attacker activity and sends alerts to security personnel. It can capture attacker's IP addresses, commands, and payloads to identify attack patterns.

Ciscoasa is a honeypot that emulates the Cisco Adaptive Security Appliance software and hardware to detect and track attacks targeting Cisco ASA devices.

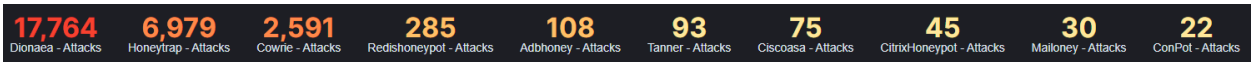
Citrix honeypot is a honeypot designed to emulate a vulnerable Citrix server environment and capture attacker behavior and tactics.

Mailoney is a honeypot that emulates a vulnerable email server environment.

Conpot is a low-interaction honeypot designed to emulate industrial control systems and SCADA protocols.

Overall T-Pot Statistics

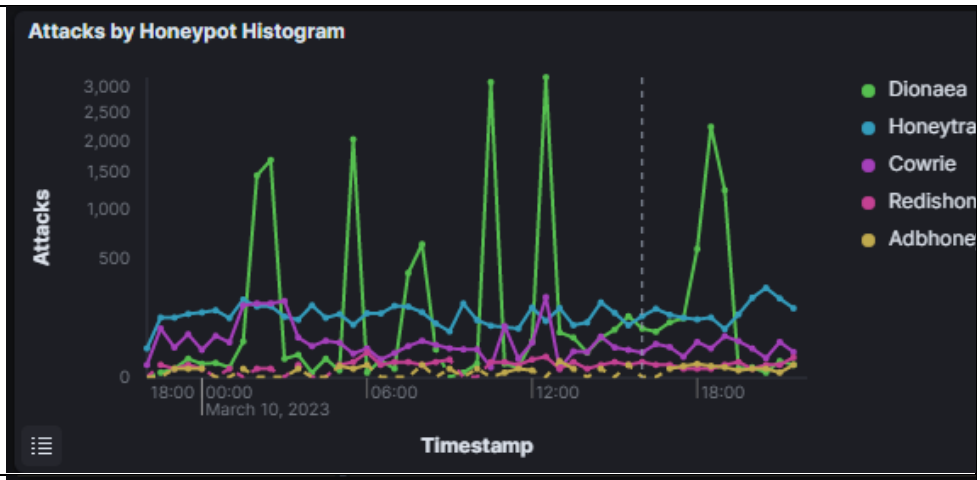
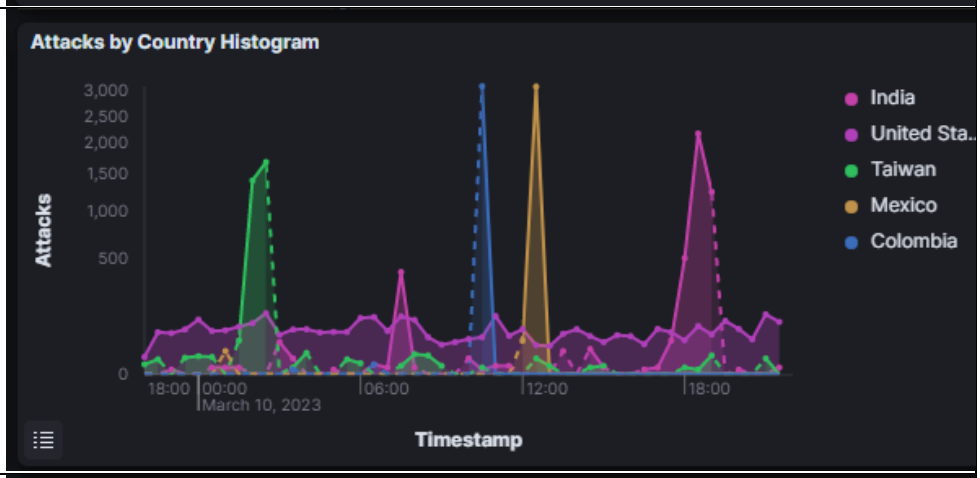
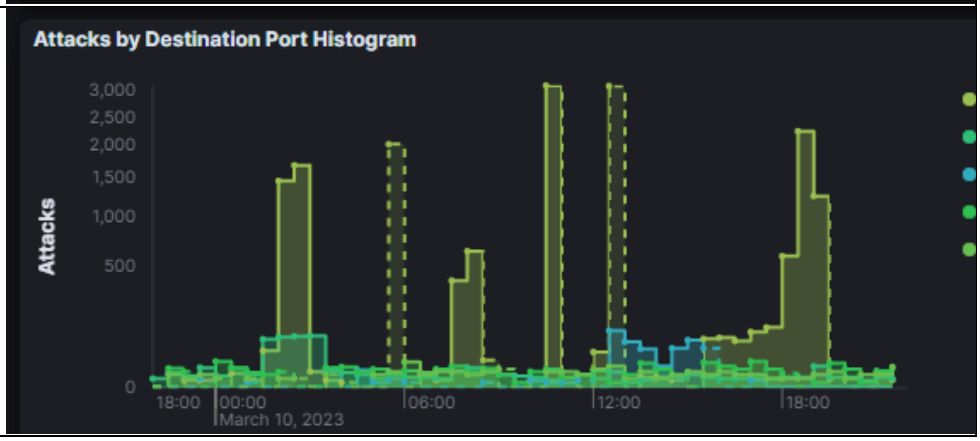
In this report we'll be covering the top three most attacked honeypots, Dionaea, honeytrap, and cowrie, but first, overall T-Pot statistics:

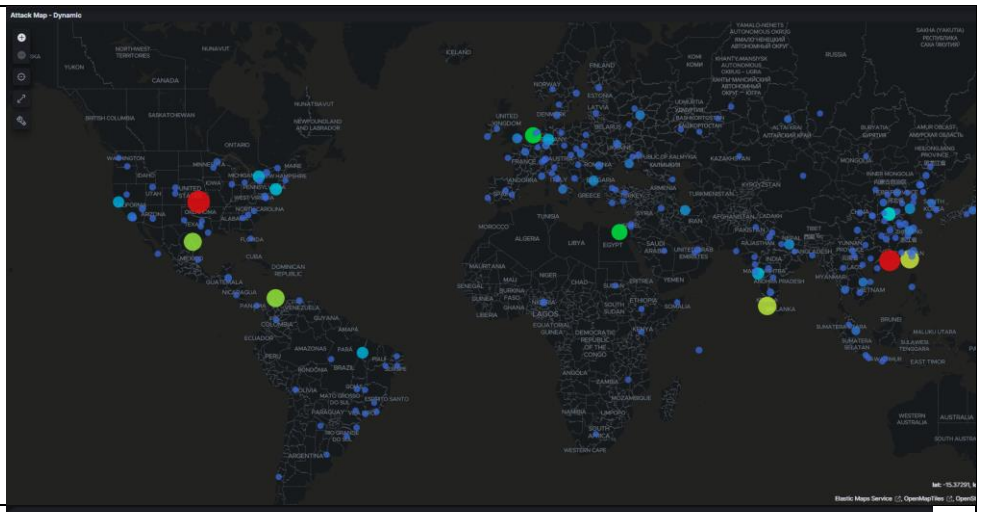
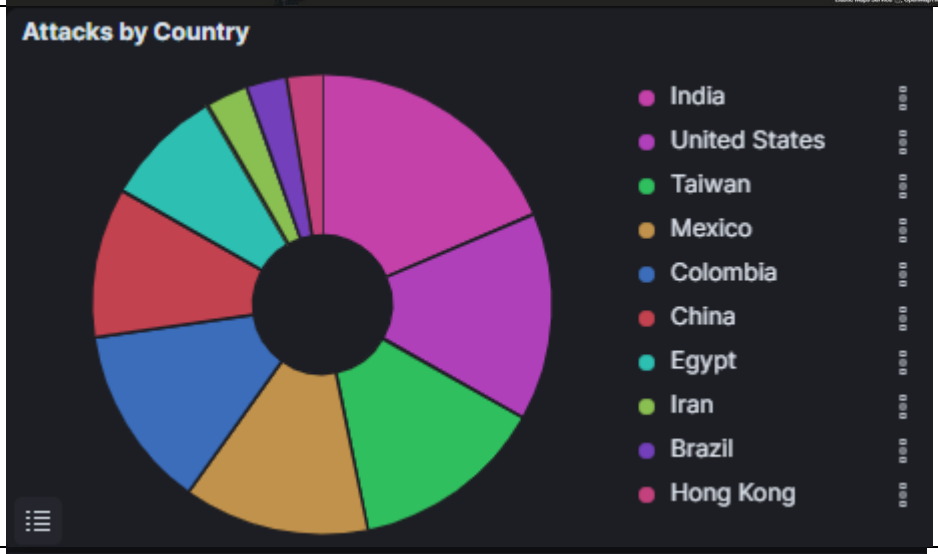
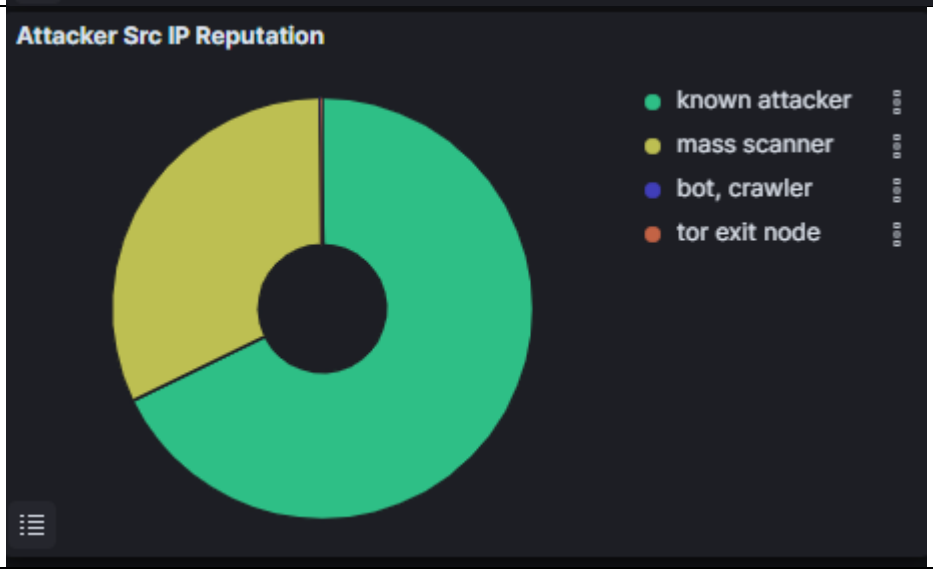


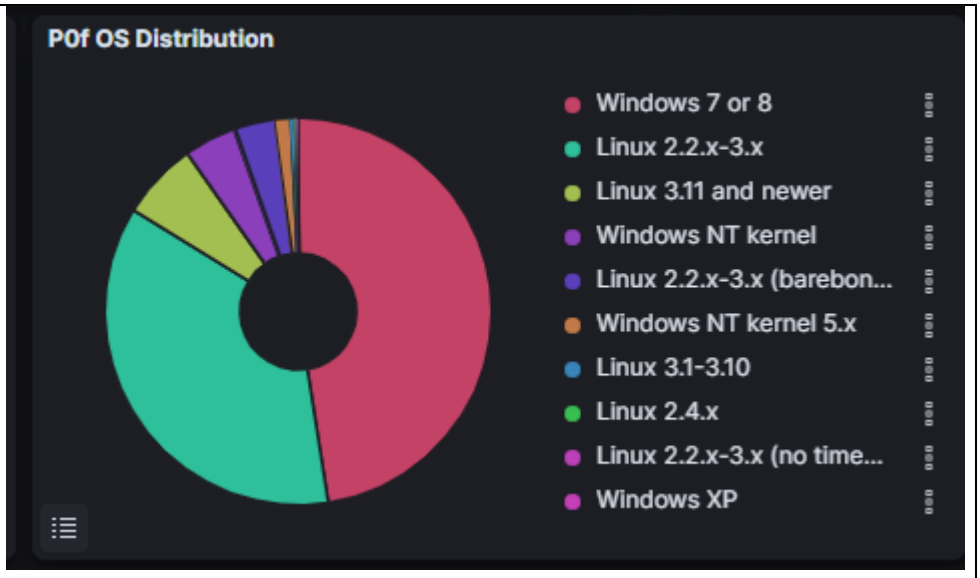
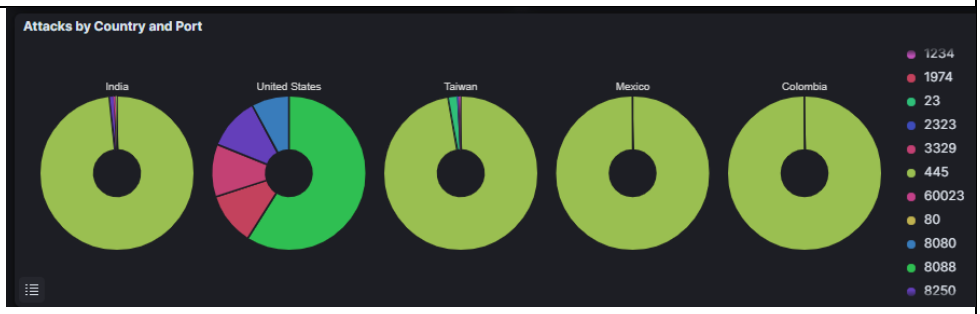
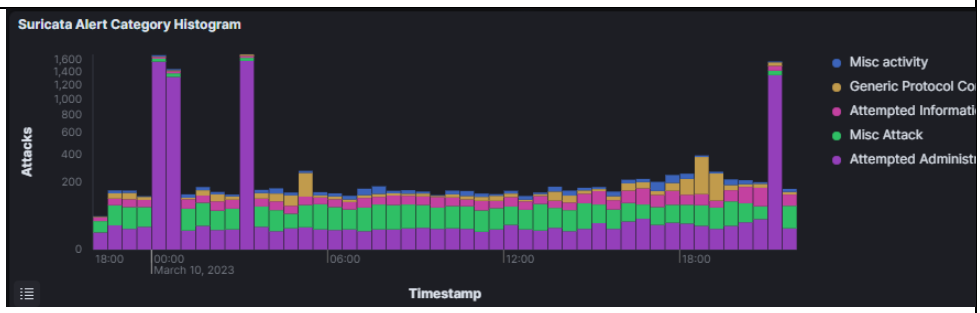
We can see that a majority of the attacks (17764) were against the Dionaea honeypot (64%), followed by the Honeytrap with 6,979 attacks (25%), and 2,591 on the cowrie honeypot (9%). Dionaea is low-interaction and emulates a Windows environment so it makes sense that it would receive a majority of the attacks from scanners or bots, especially with some of the CVE's exploited.

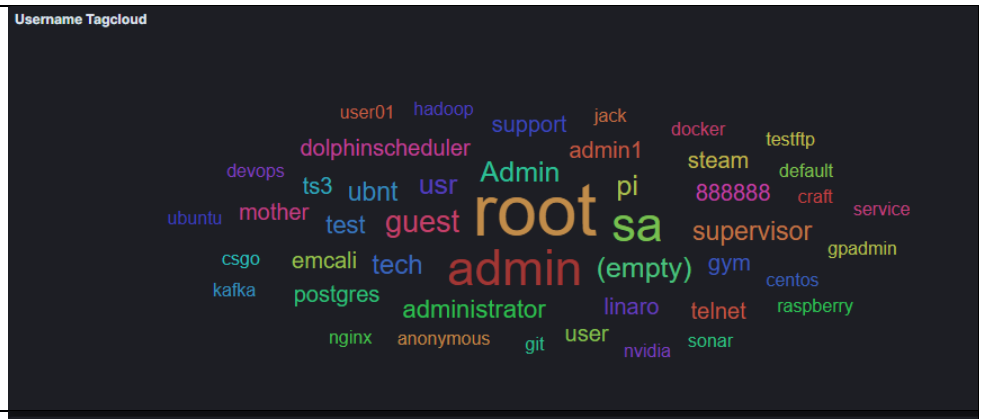
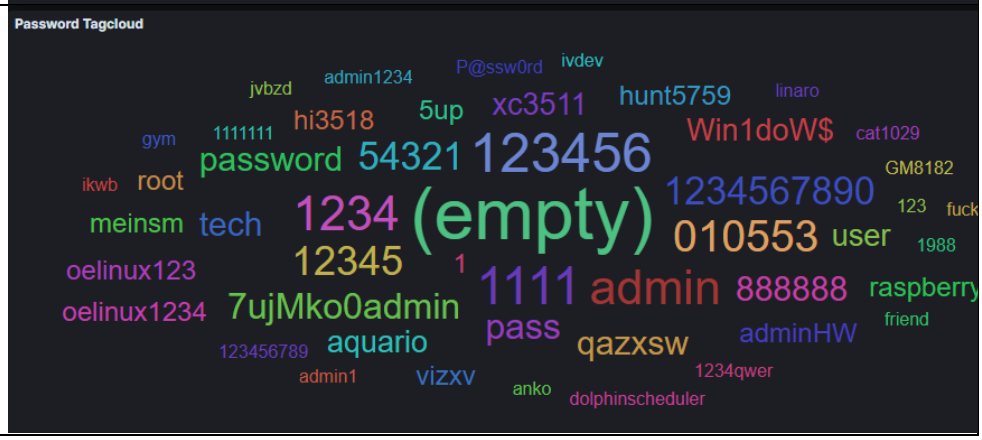
Overall STAT T-POT Board

Analysis:	Diagram:

During this time period, the most attacks spiked at 2:00-2:30, 5:30, 8:30, 10:30, 12:30, and 18:30. Most attacks appear to occur during the first half of the day. Dionaea attacks are by far the most numerous, but also appear in bursts, while honeytrap is more constant.	
The large spikes in attacks also appear to correlate with certain countries. Although the USA seems to be the most consistent with the volume of honeytrap attacks, Taiwan, Colombia, Mexico, and India seem to “spike” in attacks, surging at certain times and dying back down just as quickly.	
Port 445, which is commonly used for HTTPS traffic, appears to be the most targeted port based on the available data. This is followed by port 23, which is commonly used for Telnet, and ports 1433, 8088, and 6379.	

We can see that attacks are primarily clustered around the Eastern half of the US, Mexico/Central America, Taiwan/China, southeast Asia and India.	
We can see here the top 10 countries by percentage of attacks. India: 19% US: 15% Taiwan: 14% Mexico: 13% Columbia: 13% China: 11% Egypt: 8% Iran: 3% Brazil: 3% Hong Kong: 3%	
It appears that most attackers are either known attackers or mass scanners, or at least the ones that have an identifiable reputation.	

Some of the most commonly used OS distributions by attackers are Windows 7 or 8 (48%), Linux 2.2.x-3.x (36%), and Linux 3.11 and Windows NT (7% and 4%, respectively). Attackers may choose these OSes to exploit vulnerabilities that have not yet been patched or updated on those systems. These older systems may also be more prevalent in certain environments, particularly those with limited resources or technical expertise. Additionally, some attackers may use these systems as a means of evading detection by security measures that are designed to identify and block attacks originating from more current and widely used operating systems.	
Attacks from India, Taiwan, Mexico, and Colombia mostly target ports 445 and https, while the US is an outlier with a mix of ports 8088, 1974, 3329, 8250, and 8080 being attacked the most. These unique ports may be targeted due to their use in commonly deployed applications, such as 8080 in web servers and 3329 in remote desktop protocols.	
While the alerts of Misc activity, Generic Protocol Command Decode, Attempted Information Leak, and Misc Attack remain relatively stable and low, Attempted Administrator Privilege attacks spike around 1-2 am, 3 am and 9 pm. These attacks could be counting on employees being out of the office to try to gain access to administrator privileges, and remain unnoticed, hence why activity during regular business hours is relatively quiet.	

These are the most used usernames to try to gain access to my instance. The usernames used by attackers tend to be predictable and commonly used or are vulnerable to dictionary attacks. Consequently, it is imperative to adopt strong and unique usernames to enhance the resilience of the system against unauthorized access attempts.	
Interestingly, the most used password is simply no password at all, maybe hopefully guessing that the server has no password. This is followed by the usual top 10 passwords like password or 123456 or admin, root, or user, highly predictable and easily cracked.	

The Top CVE's Exploited Were:

CVE-2020-11899: Found in the Windows Graphics Device Interface (GDI) component, this vulnerability allows an attacker to execute arbitrary code on a targeted system by tricking a user into opening a specially crafted image file.

CVE-2001-0540: This vulnerability is a buffer overflow found in the Solaris telnet daemon that allows remote attackers to execute arbitrary code with the privileges of the telnet server.

CVE-2012-0152: In the Oracle Database Server, this vulnerability allows remote attackers to execute arbitrary code via a specially crafted SQL statement.

CVE-2002-0013: This security flaw is present in the Microsoft Windows OS, and allows attackers to execute arbitrary code via a specially crafted print request.

CVE-1999-0265: A security vulnerability found in the Linux kernel that allows attackers to obtain root privileges on a targeted system by exploiting a buffer overflow in the "ptrace" system call.

CVE-2019011500: Found in the Drupal content management system, this vulnerability allows remote attackers to execute arbitrary code via a specially crafted request.

Source: https://www.cvedetails.com/

Note: **Suricata** is a free and open-source intrusion detection system (IDS) and intrusion prevention system (IPS) developed by the Open Information Security Foundation (OISF). It is designed to monitor network traffic and detect and prevent a wide range of cyber threats including malware, viruses, and other malicious activities, and in the case of the honeypot, detect when CVEs are exploited.

Top 10 IP Addresses	ASN	Count	City/State	Country	ISP
187.254.21.24	16960	3155	Nuevo Laredo/Tamaulipas	Mexico	Cablevision Red S.A de C.V
60.249.18.240	3462	3150	Taichung/Taichung	Taiwan	Chunghwa Telecom Co Ltd
202.148.58.16	24186	3148	Maharashtra	India	Railtel Corporation of India Ltd
186.1.181.186	27837	3124	Santa Marta/Magdalena	Columbia	Dialnet de Colombia S.A E.s.p
197.47.154.15	8452	2027	Cairo/Al Qahirah	Egypt	TE-AS
103.35.132.132	134254	842	Pune/Maharashtra	Inda	Shah Infinite Solutions PVT Ltd
85.185.210.231	58224	720	Abhar/Zanjan	Iran	Iran Telecommunication Company PJS
177.185.136.216	52936	661	Imperatriz/Maranhao	Brazil	Isotelco LTDA
157.230.48.117	14061	422	North Bergen/New Jersey	US	Digitalocean LCC
223.80.102.184	24444	419	Qingdao/Shandong	China	China Mobile Communications Corporation

Parsing the Top 10 IP Address data, I was able to determine the ASN, Geolocation and ISP of these IP Addresses. It appears that most of the attacker IPs originated from Mexico, Taiwan, India or Columbia. Now we don't know for sure that these are the real addresses as they could have been spoofed, but locations such as these make sense as they are generally countries composed of lower income workers, and scanning the internet for open servers and ransoming them would be a very lucrative business to them. There also aren't as many regulations and law enforcement in these areas to combat these attackers.

Suricata Alert Signature - Top 10

ID	Description	Count
2024766	ET EXPLOIT [PTsecurity] DoublePulsar Backdoor installation communication	5,694
2402000	ET DROP Dshield Block Listed Source group 1	1,701
2009582	ET SCAN NMAP -sS window 1024	1,193
2030387	ET EXPLOIT Possible CVE-2020-11899 Multicast out-of-bound read	738
2210051	SURICATA STREAM Packet with broken ack	612
2023753	ET SCAN MS Terminal Server Traffic on Non-standard Port	574
2002752	ET POLICY Reserved Internal IP Traffic	499
2017515	ET INFO User-Agent (python-requests) Inbound to Webserver	450
2210037	SURICATA STREAM FIN recv but no session	250
2034857	ET HUNTING RDP Authentication Bypass Attempt	160

ET EXPLOIT [PTsecurity] DoublePulsar Backdoor installation communication: The most prevalent attack signature is that of DoublePulsar Backdoor, or the infamous EthernalBlue exploit, which was used in the WannaCry ransomware attack. The EternalBlue exploits a vulnerability in Microsoft Windows systems to install a backdoor on the target system, allowing them to maintain persistent access and control over the compromised machine. It's therefore no surprise that this is the most used exploit as so many Windows machines were susceptible to compromise.

ET DROP Dshield Block Listed Source group 1: A rule that blocks traffic originating from IP addresses on a known block list maintained by DShield.

ET SCAN NMAP -sS window 1024: A signature used to detect port scanning activity, specifically those conducted with the Nmap tool using a TCP SYN scan (-sS) with a window size of 1024.

ET EXPLOIT Possible CVE-2020-11899 Multicast out-of-bound read: Used to detect attempts to exploit a vulnerability in Windows Hyper-V that could allow a remote attacker to execute arbitrary code.

SURICATA STREAM Packet with broken ack: Detects TCP packets with invalid acknowledgement (ACK) numbers, which could indicate attempts to disrupt network connections or conduct denial-of-service attacks.

ET SCAN MS Terminal Server Traffic on Non-standard Port: Detects attempts to scan for Microsoft Terminal Server traffic on non-standard ports.

ET POLICY Reserved Internal IP Traffic: A rule that detects network traffic containing reserved IP addresses, which could indicate attempts to probe or exploit internal networks.

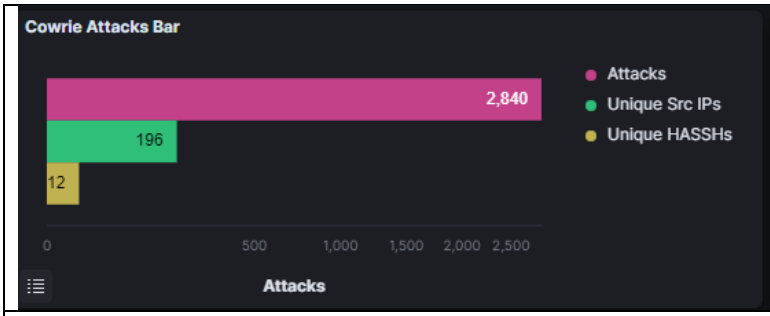
ET INFO User-Agent (python-requests) Inbound to Webserver: Notes inbound web traffic with a specific user-agent string indicating the use of the Python requests library.

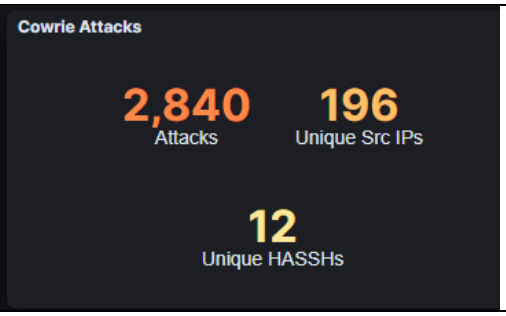
SURICATA STREAM FIN recv but no session: A rule that detects TCP packets with the FIN flag set but no corresponding session, which could indicate attempts to disrupt network connections or conduct denial-of-service attacks.

ET HUNTING RDP Authentication Bypass Attempt: A signature used to detect attempts to bypass authentication mechanisms in Remote Desktop Protocol (RDP) connections, which could allow unauthorized access to remote systems.

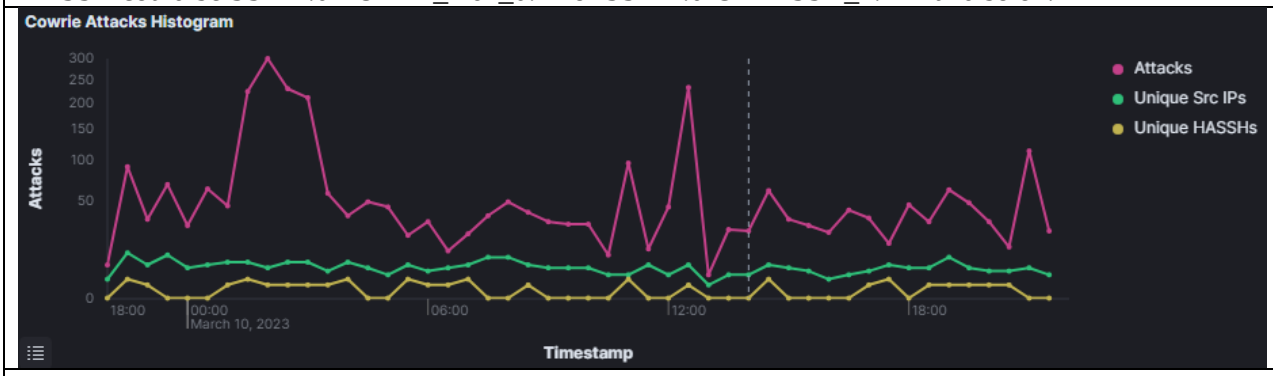
Although not the most attacked honeypot, I think Cowrie offers the most insight into the attackers motives and behavior.

Cowrie:

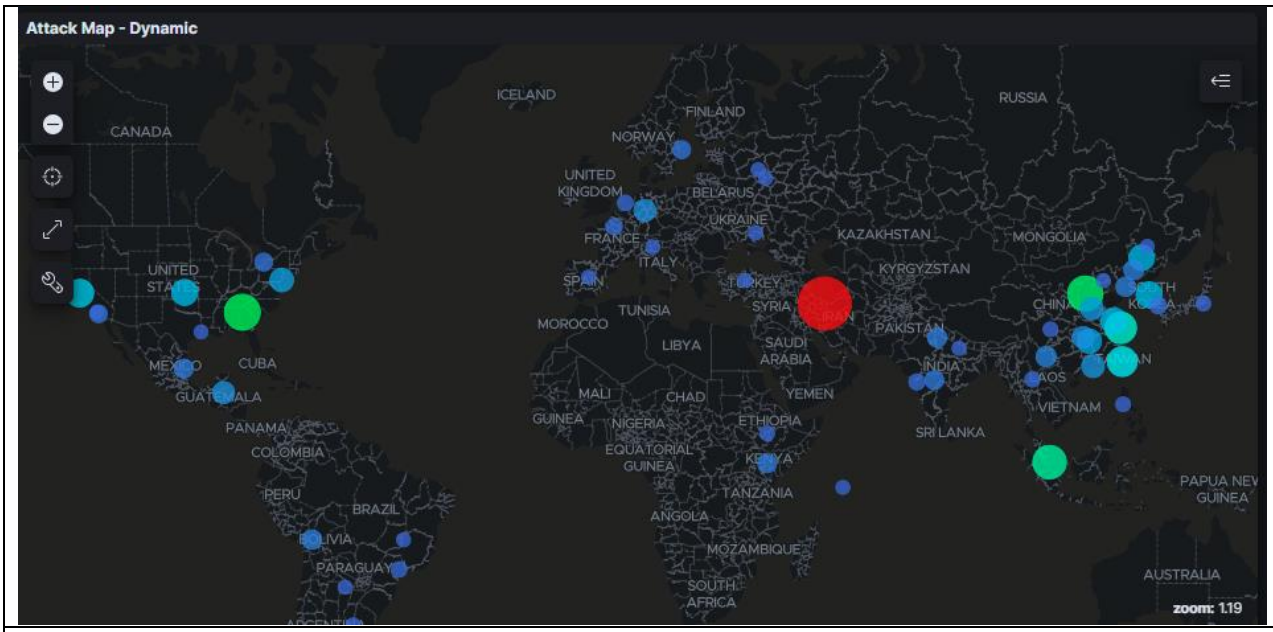




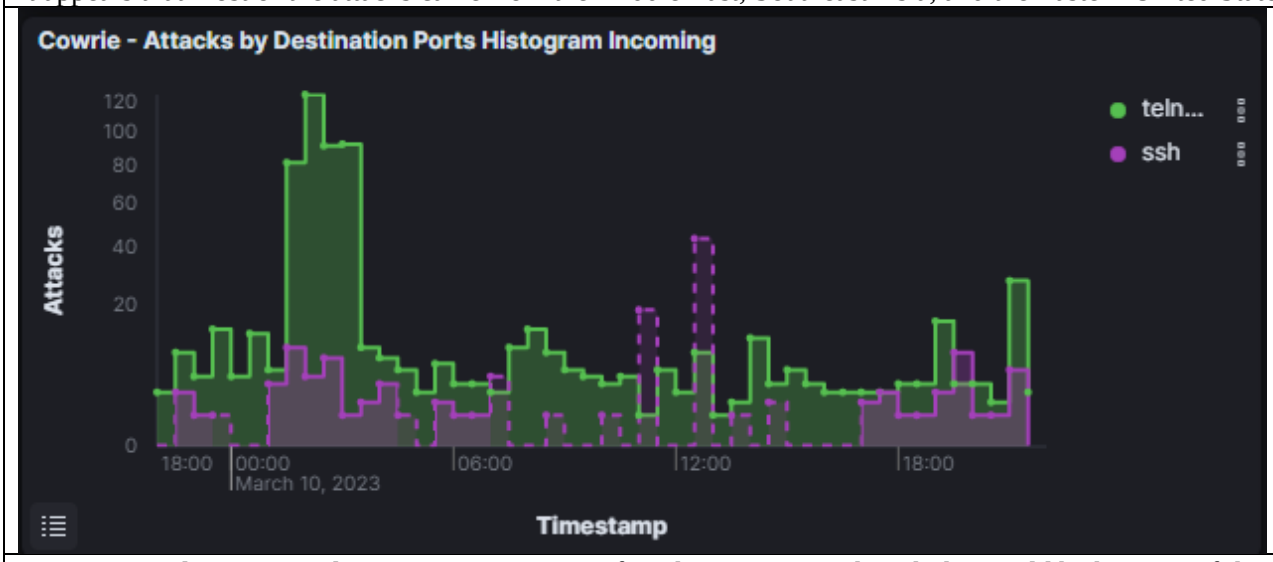
Overall, there were 2840 attacks, 192 unique IPs and 12 unique HASSHs. HASSHs or Hashes for Active SSH are fingerprinting techniques used to identify SSH clients based on unique characteristics of their traffic. For example a unique HASSH could be SSH-2.0-PUTTY_X64_0.74 or SSH-2.0-OPENSSH_7.4P1 and so on.



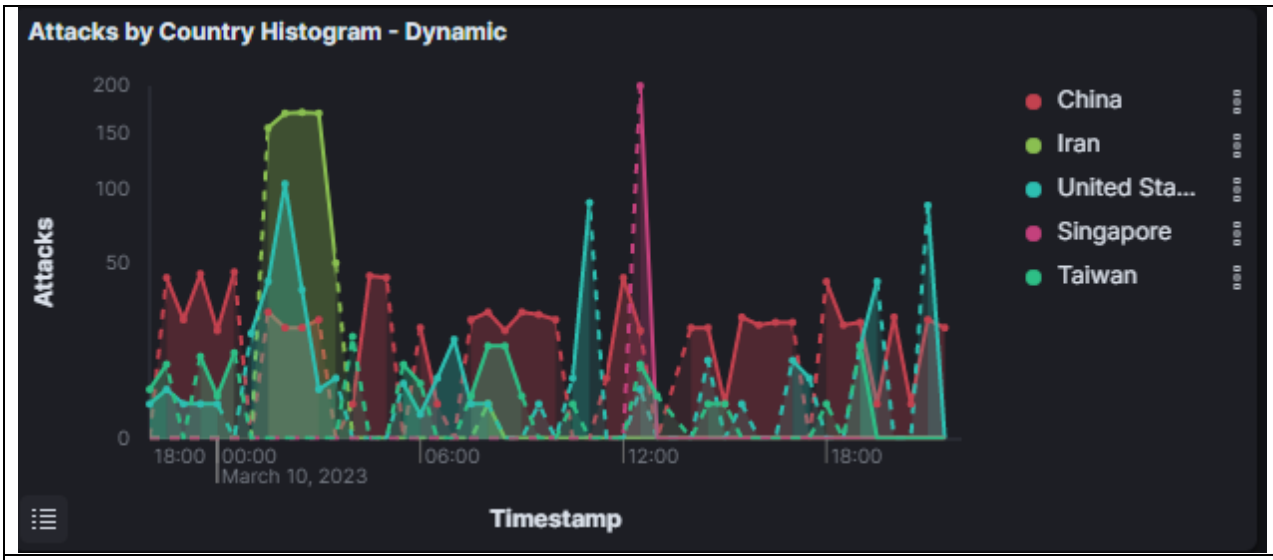
These attacks were remained mostly constant throughout the day, with attacks peaking at 3 am and noon.



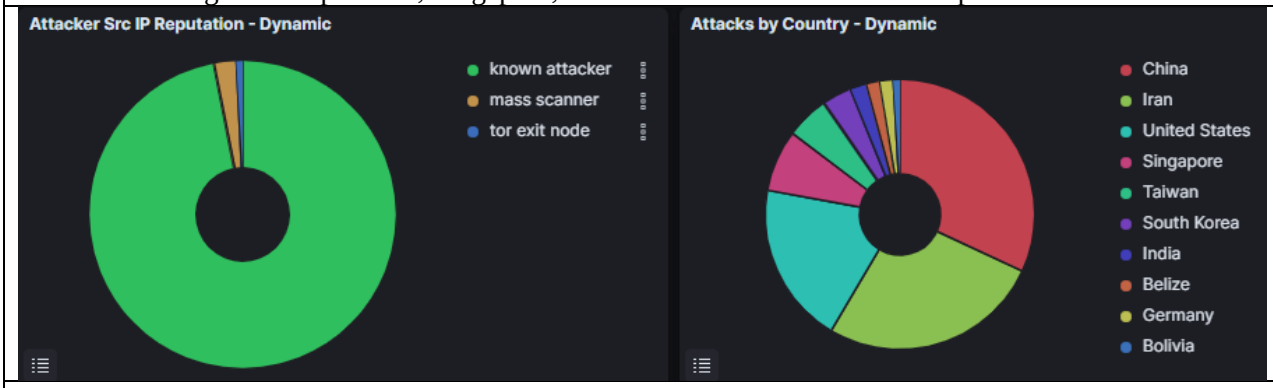
It appears that most of the attacks came from the Middle East, Southeast Asia, and the Eastern United States.



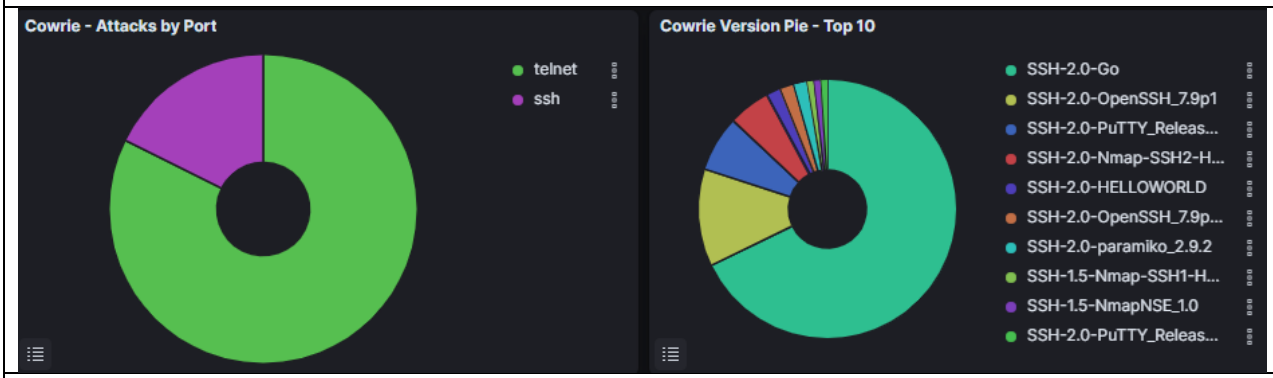
Interesting Telnet was used to connect way more often then SSH was, though that could be because of the inherit insecurity and plaintext nature of Telnet connections meaning they're less traceable.



As noted in the global map earlier, Singapore, China and Iran are some of the top countries.



Here we can see a majority of the IPs are known attackers, which I assume is because they're from questionable countries like China or Iran, which made up the majority of Cowrie attacks.
China: 34%
Iran: 29%
US: 11%
Singapore: 8%
Taiwan: 6%
South Korea: 4%



Here we can see the attacks on telnet at 84% and ssh at 16%. Also we can see what SSH/Telnet version the attackers used. Interestingly SSH-2.0-Go makes up about 78% of the attackers, SSH-2.0-OpenSSH makes up 14%, and the rest make up 18%.

So why use SSH Go?
Attackers may use SSH-2.0-Go for instigating cyber attacks over other SSH clients such as OpenSSH, PuTTY, or nmap for several reasons:

1. Evading detection: SSH-2.0-Go is not a well-known SSH client and is less likely to be detected by network security tools that look for known SSH clients. Attackers may use this client to evade detection and bypass security measures.

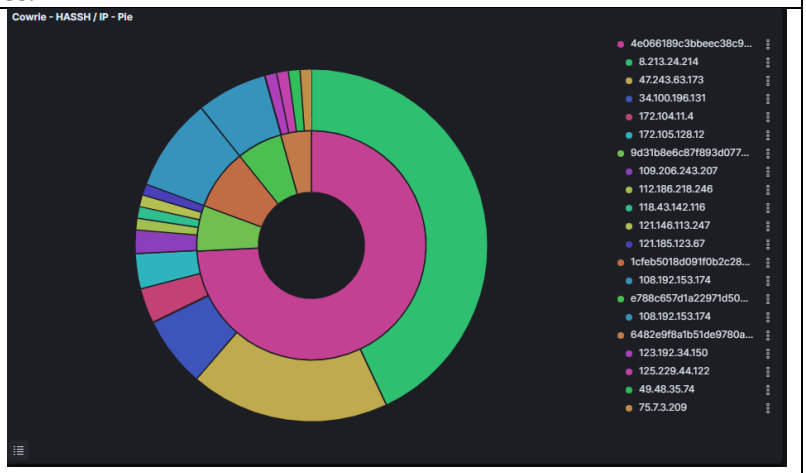
2. Customization: SSH-2.0-Go is written in Go programming language and is open-source, which allows attackers to customize the client to their specific needs. This can include modifying the behavior of the client, adding additional functionality, or incorporating new attack vectors.

3. Flexibility: Go is a lightweight and efficient programming language that allows for easy integration with other tools and libraries. Attackers may choose to use SSH-2.0-Go because it provides greater flexibility in terms of the types of attacks that can be executed.

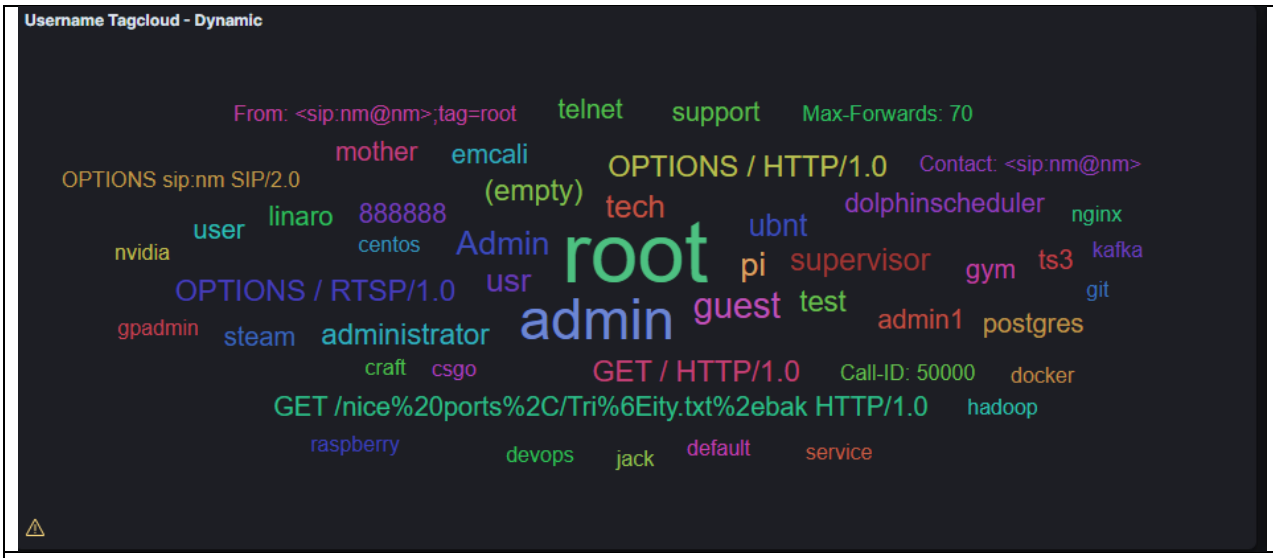
4. Performance: SSH-2.0-Go is designed for high-performance and can handle large volumes of traffic, which can be beneficial for attackers trying to conduct high-volume attacks.

Overall, attackers may use SSH-2.0-Go for instigating cyber attacks over other SSH clients because it provides them with greater flexibility, customization options, and performance.

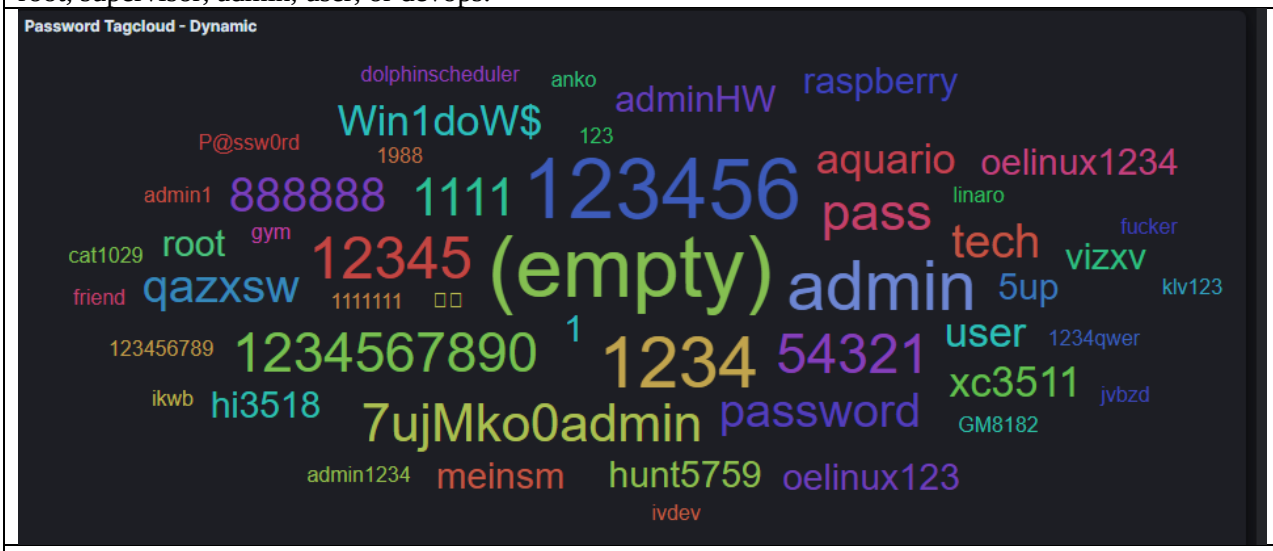
HASSH	Source IP	Count
4e066189c3bbeec38c99b1855113733a	8.213.24.214	40
4e066189c3bbeec38c99b1855113733a	47.243.63.173	17
4e066189c3bbeec38c99b1855113733a	34.100.196.131	6
4e066189c3bbeec38c99b1855113733a	172.104.11.4	3
4e066189c3bbeec38c99b1855113733a	172.105.128.12	3
9d31b8e6c87f893d077ca6526f7c710b	109.206.243.207	2
9d31b8e6c87f893d077ca6526f7c710b	112.186.218.246	1
9d31b8e6c87f893d077ca6526f7c710b	118.43.142.116	1
9d31b8e6c87f893d077ca6526f7c710b	121.146.113.247	1
9d31b8e6c87f893d077ca6526f7c710b	121.185.123.67	1
6482e9f8a1b51de9780a573985cc04fa	123.192.34.150	1
6482e9f8a1b51de9780a573985cc04fa	125.229.44.122	1
6482e9f8a1b51de9780a573985cc04fa	49.48.35.74	1
6482e9f8a1b51de9780a573985cc04fa	75.7.3.209	1
873a5fb5fedc2d4f8638ebde4abc6cfc	162.142.125.215	1
873a5fb5fedc2d4f8638ebde4abc6cfc	167.94.138.126	1
873a5fb5fedc2d4f8638ebde4abc6cfc	167.94.145.60	1
d6729b7f24428169e981ad4840063ca5	18.209.174.13	1
d6729b7f24428169e981ad4840063ca5	50.16.166.207	1



Here we can see the distribution of HASSHs (Hashes for Active SSH) and their corresponding source IP addresses. I think based off our earlier findings we can assume that the 4e HASSH corresponds to the SSH-2.0-Go software. It's interesting to see just how many IP addresses share the same HASSH value.



Here we can see the usernames used in these attacks. There's a lot more variety in the types of usernames used, including some Get requests and html, possibly to try to perform SQL injections. Most of these usernames are still predictable like root, supervisor, admin, user, or devops.



The same story occurs for the passwords too. Most of them are predictable like 12345 or password or forms of admin, but making sure your passwords are strong and you use two factor authentication is key in securing your systems from brute force or dictionary attacks.

Source IP	Count	ASN	City/State	Country	ISP
85.185.210.231	720	58224	Abhar/Zanjan	Iran	Iran Telecommunication Company
8.213.24.214	201	45102	Riyadh/Ar Riyad	Saudi Arabia	Alibaba Technology Co Ltd
47.243.63.173	90	45102	Hong Kong	Hong Kong	Alibaba Technology Co Ltd
36.48.64.189	41	4134	Changchun/Jilin	China	Jilin Telecom Corporation
34.100.196.131	37	396982	Mumbai/Maharashtra	India	Google LLC
190.109.236.80	25	52495	La Paz	Boliva	Cotel LTDA
106.111.38.59	23	4134	Guangzhou/Guangdon	China	ChinaNet Guangdong Province Network
123.185.223.115	23	134762	Dalian/Liaoning	China	ChinaNet Liaoning Province Network
223.13.64.108	23	4134	Taiyuan/Shanxi	China	ChinaNet Shanxi Province Network
110.182.240.122	22	4134	Taiyuan/Shanxi	China	ChinaNet Shanxi Province Network

The IP addresses associated with the traffic show a concentration in certain countries like Iran, Hong Kong, China, and India. However, there are instances where IP addresses from different geographic locations share the same Internet Service

Provider (ISP), such as 8.213.24.214 and 47.243.63.173, both belonging to Alibaba Technology Co Ltd. It is unclear whether this is due to address spoofing or if Alibaba has servers located in different regions. Additionally, a significant amount of traffic originates from the Autonomous System Number (ASN) 4134, which likely represents the same ISP, but with different regional branches like ChinaNet Guangdong vs Liaoning or Shanxi.

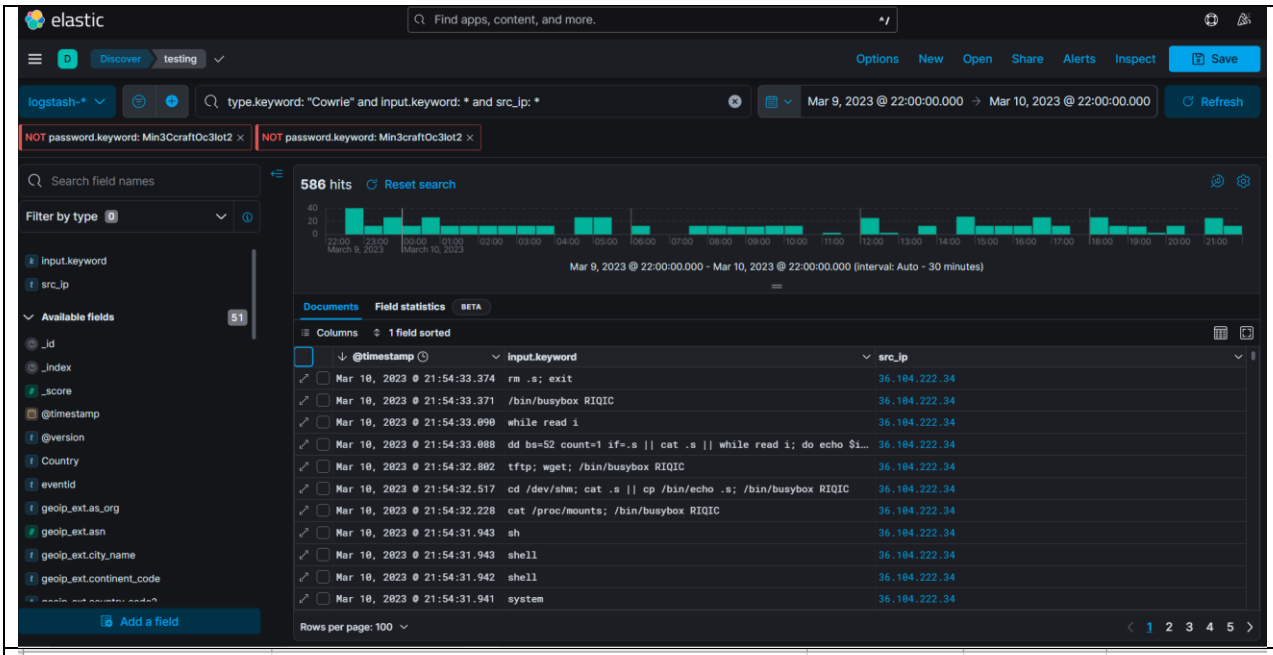
The concentration of traffic from certain countries, along with instances of IP addresses from different geographic locations sharing the same ISP, could indicate malicious actors attempting to obfuscate their origin and evade detection.

Furthermore, the use of HASSH fingerprinting and unique SSH clients highlights the importance of monitoring and analyzing network traffic for anomalous or suspicious behavior. This type of analysis can aid in the detection and mitigation of cyber threats, particularly those that may be attempting to exploit vulnerabilities or gain unauthorized access through SSH protocols.

Possible Ransomware Scripts

Cowrie Input - Top 10

Command Line Input	Count
shell	90
system	90
dd bs=52 count=1 if=.s || cat .s || while read i; do echo $i; done < .s	45
enable	45
sh	45
while read i	45
rm .s; exit	38
cat /bin/echo;	2
uname -s -v -n -r -m	2
/bin/busybox AOAEM	1



@timestamp	input.keyword	src_ip	geoip.city_name	geoip.country_name
Mar 10, 2023 @ 21:54:33.374	rm .s; exit	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:33.371	/bin/busybox RIQIC	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:33.090	while read i	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:33.088	dd bs=52 count=1 if=.s || cat .s || while read i; do echo $i; done < .s	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:32.802	tftp; wget; /bin/busybox RIQIC	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:32.517	cd /dev/shm; cat .s || cp /bin/echo .s; /bin/busybox RIQIC	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:32.228	cat /proc/mounts; /bin/busybox RIQIC	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:31.943	sh	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:31.943	shell	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:31.942	shell	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:31.941	system	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:31.940	system	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:54:31.938	enable	36.104.222.34	Changchun	China
Mar 10, 2023 @ 21:12:08.421	rm .s; exit	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:08.418	/bin/busybox WPUKO	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:08.178	while read i	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:08.175	dd bs=52 count=1 if=.s || cat .s || while read i; do echo $i; done < .s	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:07.932	tftp; wget; /bin/busybox WPUKO	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:07.691	cd /dev/shm; cat .s || cp /bin/echo .s; /bin/busybox WPUKO	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:07.448	cat /proc/mounts; /bin/busybox WPUKO	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:07.208	sh	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:07.207	shell	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:07.206	shell	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:07.206	system	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:07.205	system	149.113.219.137	-	United States
Mar 10, 2023 @ 21:12:07.202	enable	149.113.219.137	-	United States
Mar 10, 2023 @ 21:05:23.524	/bin/busybox LDWDD	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:23.323	while read i	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:23.321	dd bs=52 count=1 if=.s || cat .s || while read i; do echo $i; done < .s	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:23.109	tftp; wget; /bin/busybox LDWDD	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:22.907	cd /dev/shm; cat .s || cp /bin/echo .s; /bin/busybox LDWDD	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:22.696	cat /proc/mounts; /bin/busybox LDWDD	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:22.492	sh	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:22.491	shell	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:22.490	shell	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:22.489	system	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:22.488	system	121.224.132.3	Suzhou	China
Mar 10, 2023 @ 21:05:22.486	enable	121.224.132.3	Suzhou	China

We can see in Excel they're following the same script:
enable, system, system, shell, sh, cat /proc/mounts; /bin/busybox RIQIC, cd/dev/shm; cat.s || cp/bin/echo.s;/bin/busybox RIQIC, tftp;wget;/bin/busyboxRIQIC, dd bs=52 count=1 if=.s || cat.s || while read I;do echo$i;done<.s, while read I, /bin/busybox RIQIC, rm.s;exit

Now what do each of these commands do? And what are the attackers trying to accomplish?

Since Cowrie is a shell emulating a UNIX system in python, its command line likely looks like user@hostname:~$, so when a remote attacker connects to the Cowrie honeypot and provides valid login credentials, they are presented with a simulated shell prompt that mimics the command-line interface of a real OS.

What is BusyBox?

BusyBox is a software suite that provides several Unix utilities in a single executable file. It is often used in scripting attacks to execute Unix commands and perform malicious activities such as privilege escalation, data exfiltration, and the installation of additional malware.

BusyBox is advantageous in scripting attacks due to its lightweight design and the wide range of Unix utilities it provides in a single file. Its ability to achieve privilege escalation on a target system makes it a common toolset for attackers.

Command Explanations:

#1. Enable: This command could be used to elevate the privileges of the user running the script, potentially allowing them to perform actions that they would not otherwise be able to do.

#2. **System**: The second command was *system*. The "system" command in Cowrie honeypot allows attackers who have gained access to execute shell commands on the host operating system. This can be used to perform malicious actions such as installing malware or stealing data. The command gives attackers the ability to execute arbitrary commands, which may be used to perform actions outside of the simulated environment.

#3. **Shell**: One of the most common inputs was *shell*, with 90 uses. The “shell” command is likely used to access a remote shell prompt on the honeypot system. This gives them greater control over the system and the ability to execute arbitrary commands. Basically allowing attackers to perform reconnaissance, escalate privledges, install malware (see section 7), or exfiltrate data from the honeypot system.

#4. **sh**: This command is used to invoke the Bourne shell, a standard Unix shell.

#5. **cat /proc/mounts; /bin/busybox RIQIC**: In this command, it appears the attackers cat (concatentated) the output of /proc/mounts, then used busybox’s output to find a folder they could use the dropper to upload the malware to.

#6. **cd /dev/shm; cat .s || cp /bin/echo .s; /bin/busybox RIQIC**: This command is attempting to change the current working directory to /dev/shm, which is a shared memory filesystem used by Unix systems. Once in that directory, it tries to read the contents of a file called .s. If that file does not exist, the command copies the file /bin/echo to a new file called .s. By copying /bin/echo into .s, the new file retains the permissions of the original file, and the inclusion of a dot in front of the filename ensures that the file's permissions are preserved.

#7. **tftp; wget; /bin/busybox RIQIC**: This command is attempting to see if it can download files from a remote server using the TFTP and Wget protocols. Once the files are downloaded, the command uses the /bin/busybox RIQIC command to execute them on the target system.

#8. **dd bs=52 count=1 if=.s || cat .s || while read i; do echo $i; done <.s:** This command is designed to read data from a file named ".s" using the "dd" command, which reads one block of data (52 bytes) from the file and outputs it to the terminal. If the ".s" file is not found, the command will try to output the contents of the file using the "cat" command. If both of these commands fail, the command will read the

contents of the ".s" file line by line and echo each line to the terminal using a "while" loop and the "echo" command. The purpose of this command is to read and output the contents of the ".s" file, which is likely a small executable file used to execute malicious commands or install malware on the target system.

#9. **while read i**: This command is used to read input from a stream, which could be used to process input from a file or network socket, in this case, “i”.

#10. **/bin/busybox RIQIC**: This command calls busybox.

#11. **rm.s;exit**: They removed the ELF file .s and terminated the shell connection. This suggests that the attackers were trying to cover their tracks by removing any traces of their activity on the target system and terminating the connection.

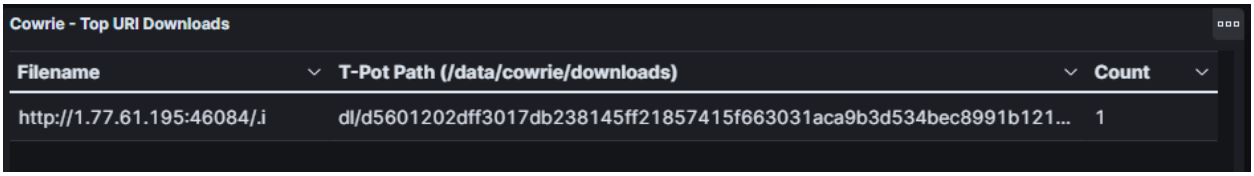
Analysis:

It seems that the attackers gained access to the system and immediately began searching for a directory to place their malware. They then attempted to modify a file named ".s" to transfer their malware onto the system, using a combination of the "cat" and "cp" commands. They also attempted to download files from a remote server using the TFTP and Wget protocols, potentially to retrieve additional tools or files to assist in their attack.

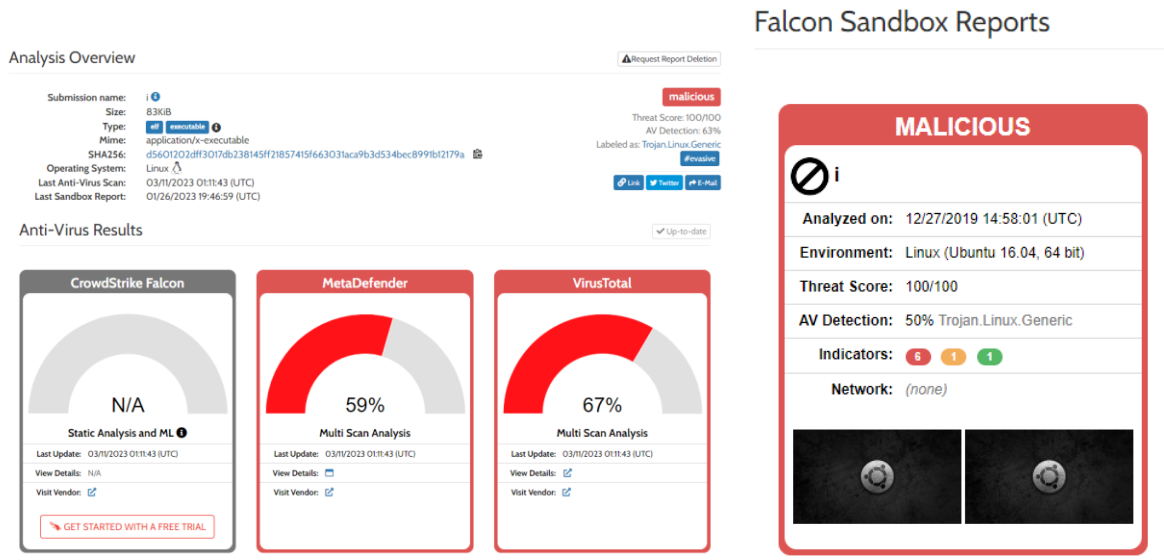
They then ran a series of commands, likely to compile or build their malware, and read the output to begin executing it on the system. They also used the "dd" command to read and output the contents of the ".s" file, which is likely a small executable file used to execute malicious commands or install malware on the target system. Finally, they removed the ".s" file and terminated the shell connection.

Overall, it appears that the attackers' goal was to gain access to the system, place their malware onto it, and execute it to achieve their objectives, then cover their tracks. The specific objectives are unclear from the provided information, but could potentially include stealing data, compromising other systems, or using the compromised system as a launching point for further attacks.

However, we may have a clue about what kind of software that busybox file had on it, since Cowire recorded a URI request to an h-t-t-p://1.77.61.195:46084/.i

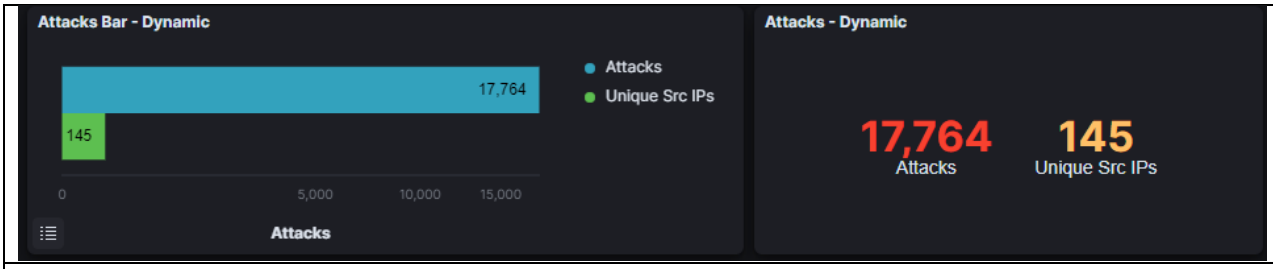


After analyzing the link using [hybrid analysis], it came back with these results.

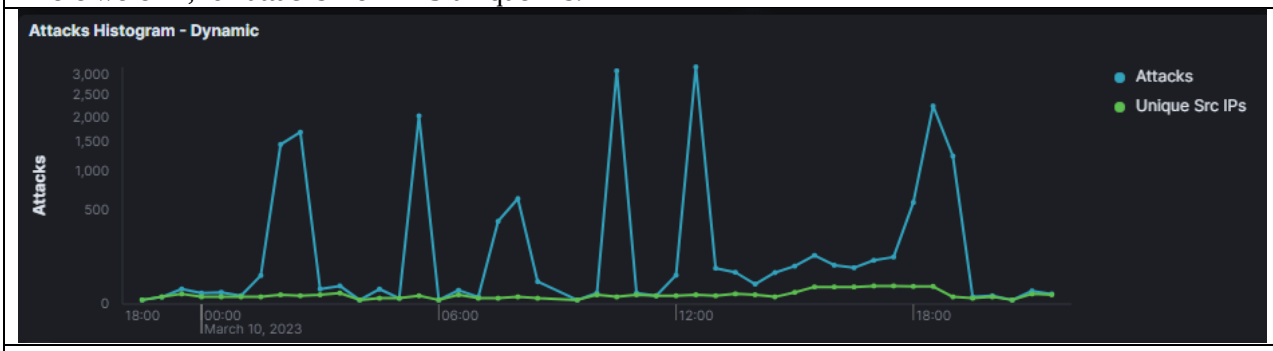


It appears to have downloaded a Trojan file, which fits with the motives and behavior of the attackers. The attacks mainly come from China or India, so installing ransomware and covering their tracks would be very profitable for them, especially in the economic downturn that occurred after COVID-19.

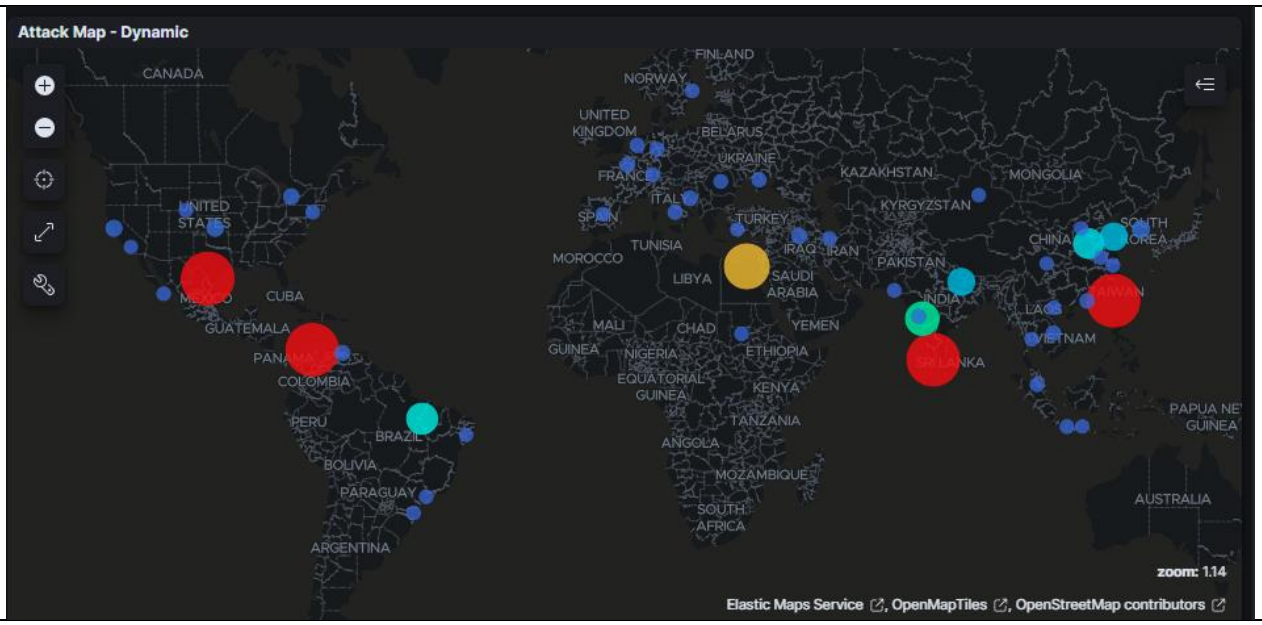
Dionaea:



There were 17,764 attacks from 145 unique IPs.

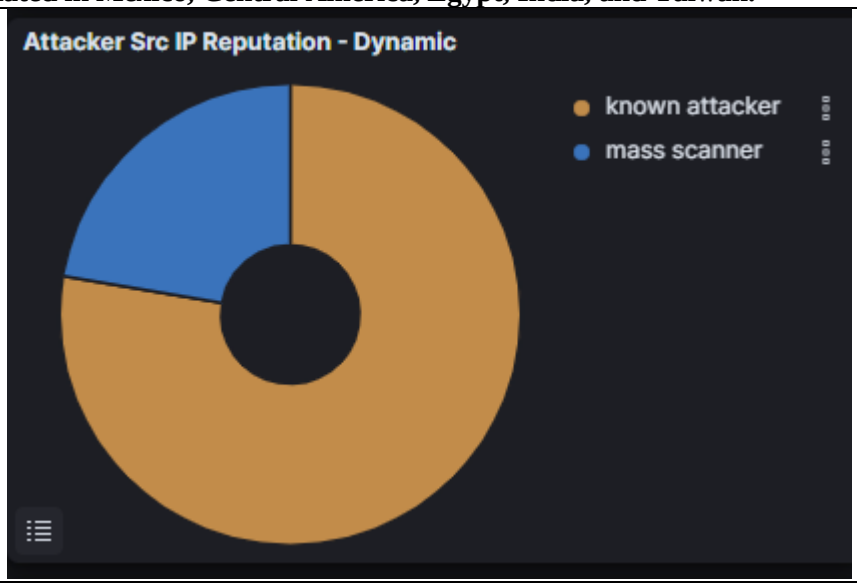


These attacks spiked at 1-3 am, 5-5:30, 7-7:30, 10-10:30, 12pm-12:30pm, and 6 pm.

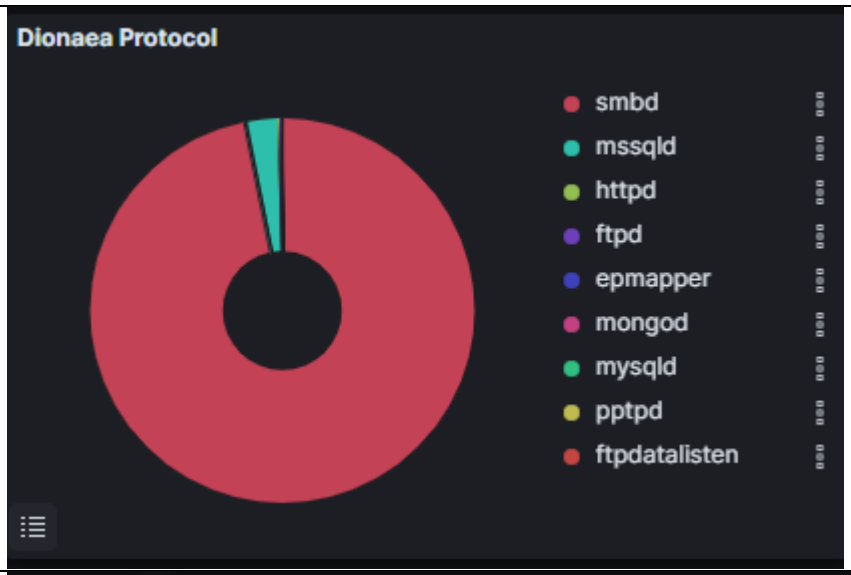


The bulk of these honeypot attacks originated in Mexico, Central America, Egypt, India, and Taiwan.

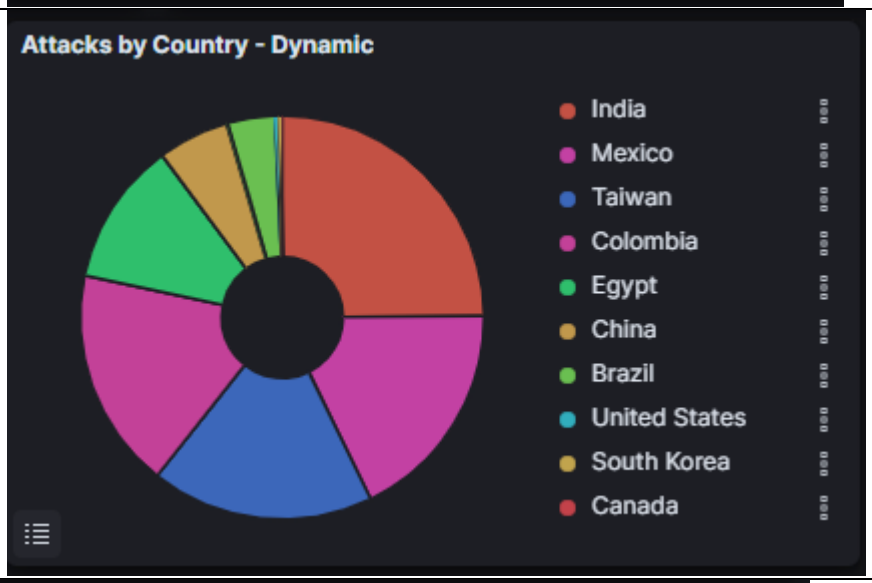
78% of these attacks were from known attackers, and 22% were from mass scanners.

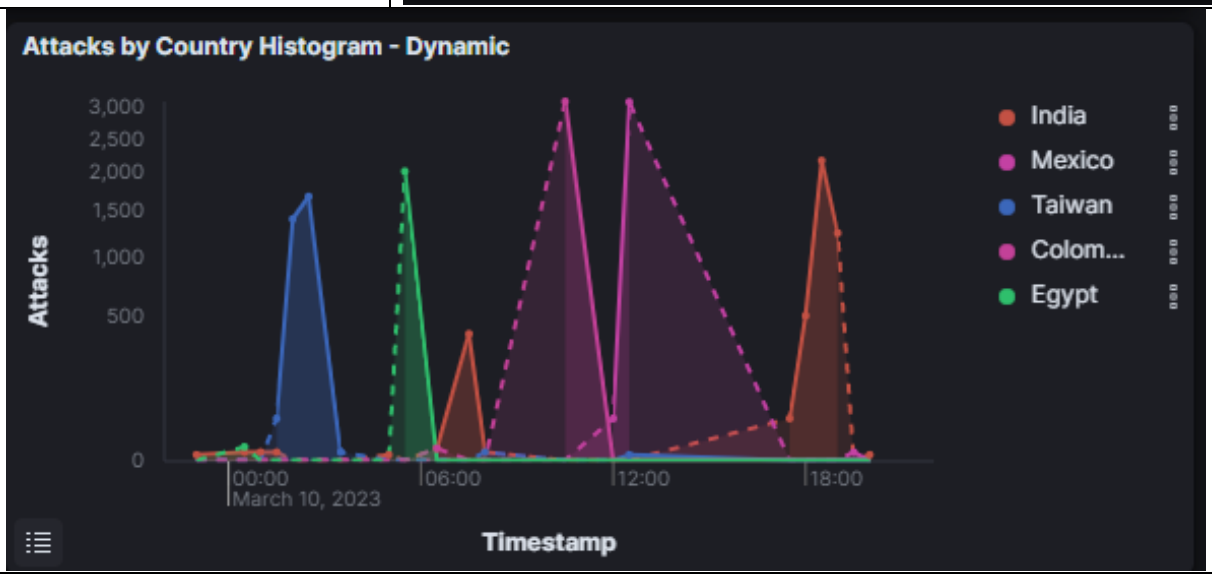


The most common protocol used was smbd, at 97%, followed by mssqld at 3%.

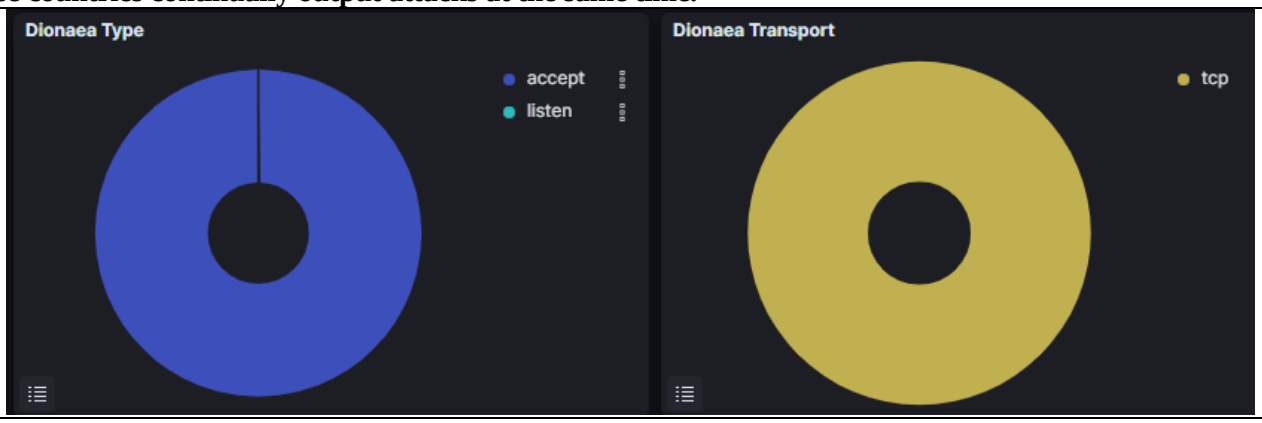


Most attacks originated from India, with 25% of the attacks:
Mexico: 18%
Taiwan: 18%
Colombia: 18%
Egypt: 12%
China: 6%
Brazil: 3%
Other: <1%



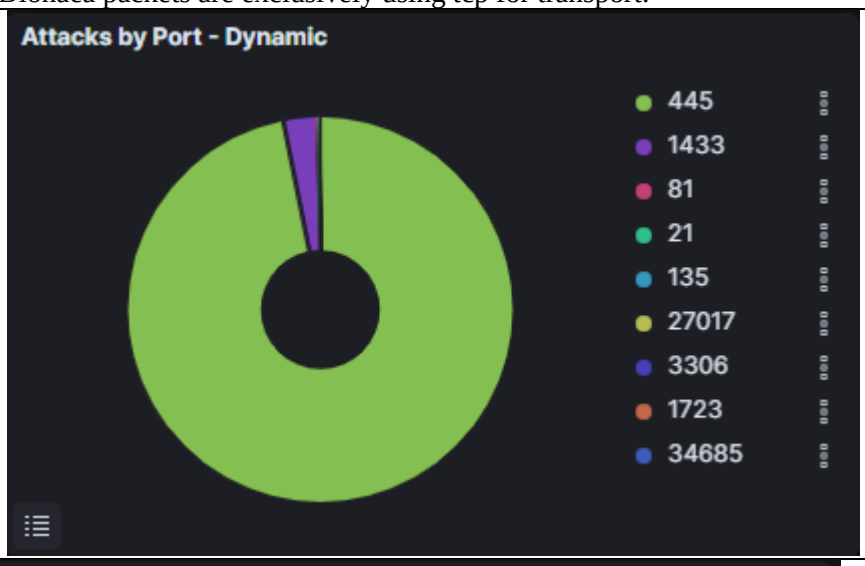


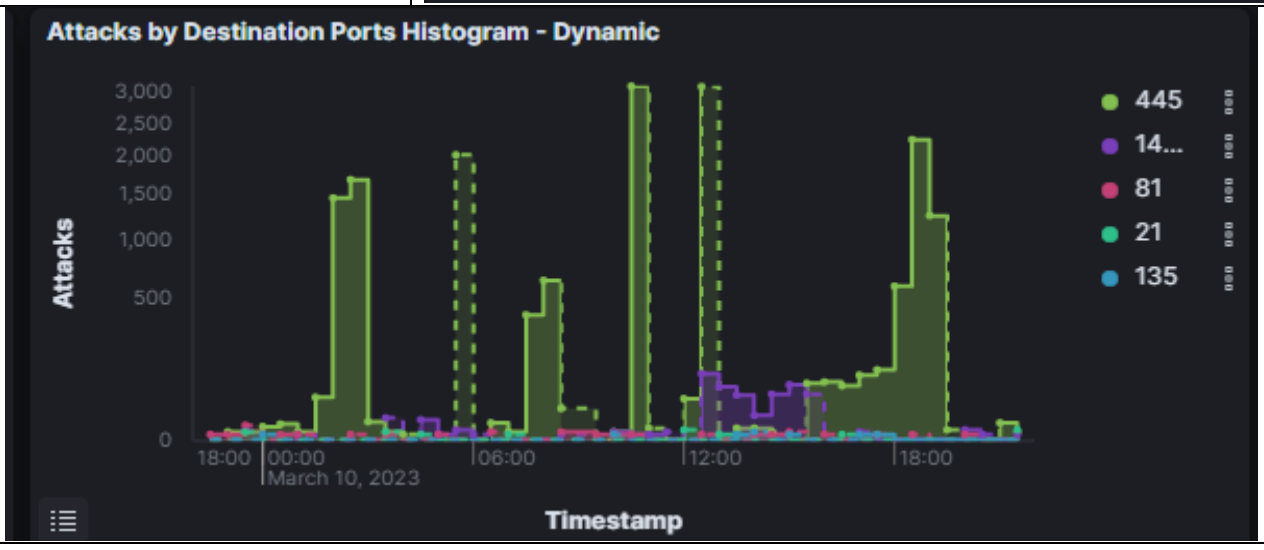
We can see that each of the spikes in attacks corresponds to a specific country, which is interesting and may show that these scans occur in waves or at specific times. Monitoring future days will determine if this is a one off trend or if these countries continually output attacks at the same time.



Dionaea “accept” and “listen” refers to how the software handles icnoming connections, with “listen” refering to waiting for and accepting incoming connections while “accept” refers to actually accepting the request. We can see here that it’s basically 100% accepting of requests. Also Dionaea packets are exclusively using tcp for transport.

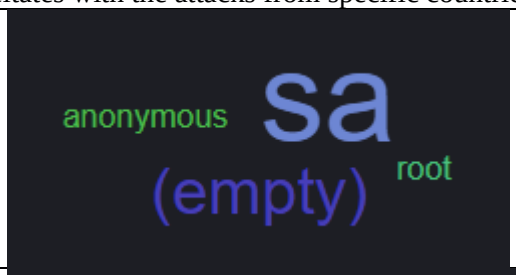
We can also see like for most of the honeypots, the port attacked is mainly 445, or https, which makes sense since the instance is hosted in the cloud.



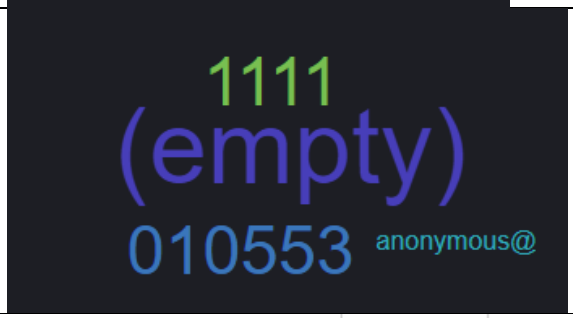


The spikes in 445 port attacks roughly correltates with the attacks from specific countries from earlier as well.

There were a lot less usernames recorded since this honeypot is not specifically geared towards that but it still includes some predictable **ones.**



For both usernames and passwords, the most prevelent was just a lack of input.



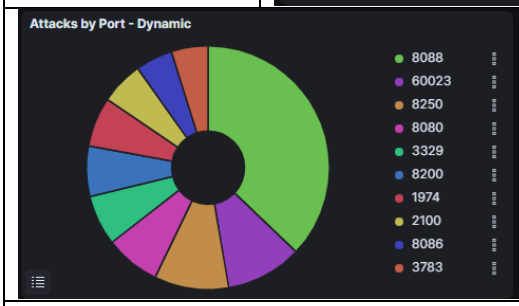
Source IP	Count	ASN	City/State	Country	ISP
187.254.21.24	3,155	16960	Nuevo Laredo/Tamaulipas	Mexico	Cablevision Red S.A de C.V
60.249.18.240	3,150	3462	Taichung/Taichung	Taiwan	Chunghwa Telecom Co Ltd
202.148.58.16	3,148	24186	Maharashtra	India	Railtel Corporation of India Ltd
186.1.181.186	3,124	27837	Santa Marta/Magdalena	Columbia	Dialnet de Colombia S.A E.s.p
197.47.154.15	2,027	8452	Cairo/Al Qahirah	Egypt	TE-AS
103.35.132.132	842	134254	Pune/Maharshtra	India	Shah Infinite Solutions PVT Ltd
177.185.136.21	661	52936	Imperatriz/Maranhao	Brazil	Isotelco LTDA
223.80.102.184	419	24444	Qingdao/Shandong	China	China Mobile
112.196.143.25	392	45184	Delhi	India	Den Networks Limited
123.55.236.133	52	4134	Luohe/Henan	China	Henan Telecom Corporation

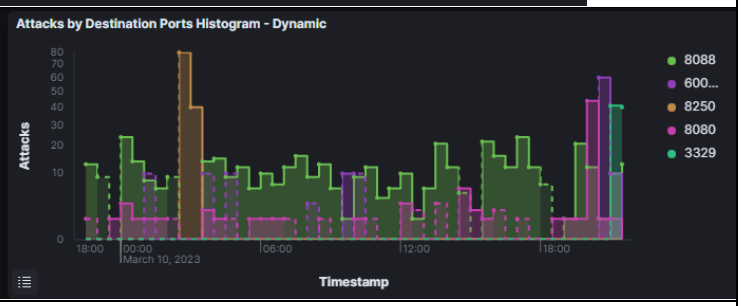
Here we can see that basically the same amount of attacks came from Mexico, Taiwan, India, and Columbia, the same culpurits that we saw in the overall T-Pot dashboard. These are definitely IPs to flag, and check if there are actually legitimate connections coming from these georgraphical regions.

Overall this honeypot didn't give me a whole lot of information that I didn't already know, which makes sense since this is a low interaction honeypot. I gleaned a lot more from the other honeypots, especially Cowrie.

HONEYTRAP

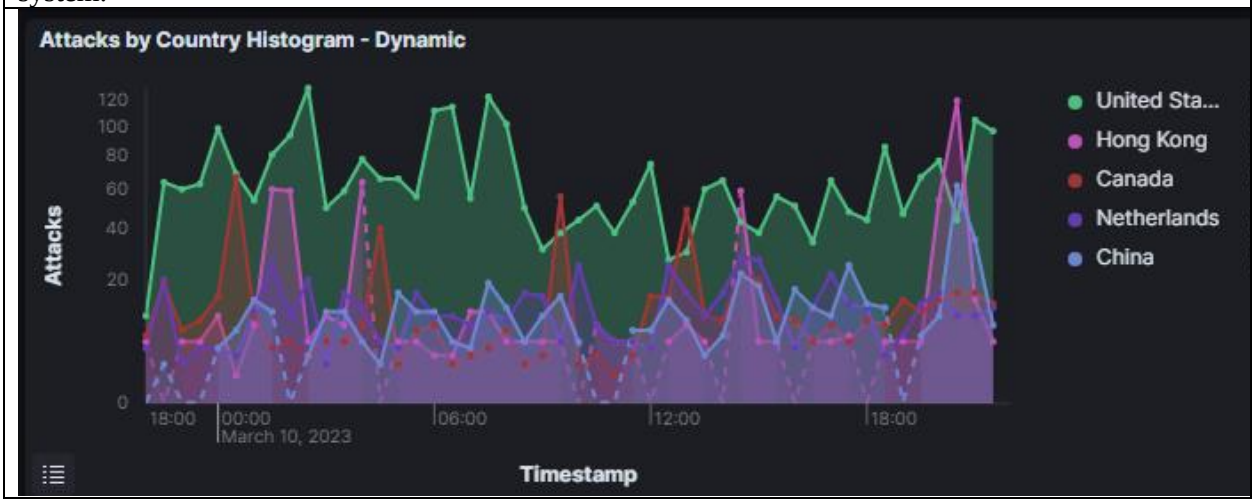
The Honeytrap honeypot had the second most attacks in the first 24 hours behind Dionaea, with 6979 attacks from 1081 unique IPs.



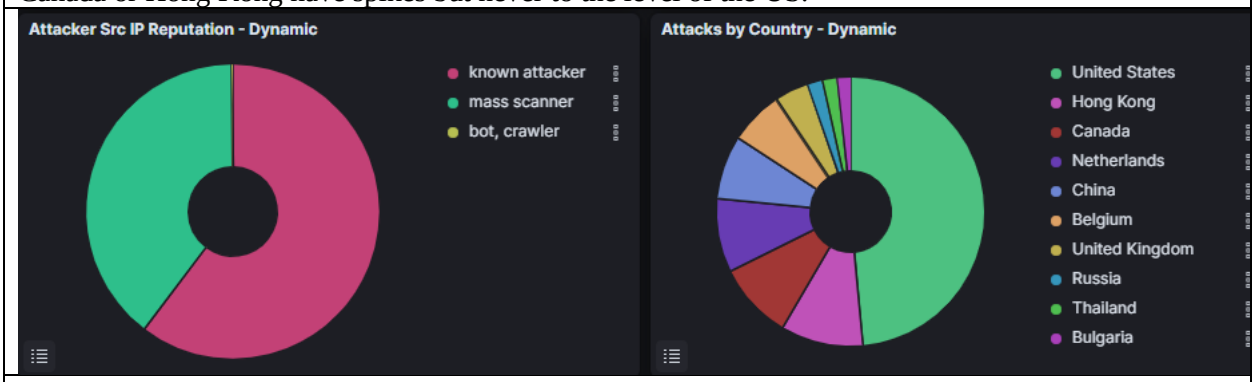


While port 8088 was the most constant port attacked, 8250 did have a major spike around 3 am. Port 8088 had 37% of the attacks, followed by 60023 at 10%, 8250 at 10%, 8080 at 7%, 8200 at 7%, and the others taking up the rest of the attacks. So why port 8088? Some explanations could be that port 8088 is commonly used for proxy servers, and attackers may attempt to exploit vulnerabilities in proxy server software to gain unauthorized access to a network or system. Additionally, some web applications may also use port 8088, and attackers may target these applications to exploit vulnerabilities and gain access to sensitive data.

Port 8250 was also likely attacked because it's commonly used by remote management software, which can provide attackers with access to sensitive data or the ability to take control of the system. Additionally, some malware and Trojans use this port to communicate with their command and control servers. Therefore, attackers often target this port to gain unauthorized access or to control the infected system.



In contrast to other honeypots, the vast majority of attacks are domestic, with the United States remaining a constant source of attacks during the entire 24 hour time period. Other countries like Canada or Hong Kong have spikes but never to the level of the US.



Adding on to that, most of the attackers are known or mass scanners, and the US takes up a whopping 49% of the attacks, followed by Hong Kong, Canada, the Netherlands, and China in equal parts.

Source IP	Count	ASN	City/State	Country	ISP
157.230.48.117	422	14061	North Bergen/New Jersey	United States	Digital Ocean LLC
164.52.54.36	175	63199		Tokyo	UCUL JP
162.142.125.216	128	398324	Ann Arbor/Michigan	United States	Censys Inc
103.210.23.31	109	135377	Hong Kong	Hong Kong	Zenlayer Inc
169.197.113.158	108	135377	London/England	Great Britain	Zenlayer Inc
162.142.125.13	100	398324	Ann Arbor/Michigan	United States	Censys Inc
162.142.125.217	91	398324	Ann Arbor/Michigan	United States	Censys Inc
162.142.125.11	81	398324	Ann Arbor/Michigan	United States	Censys Inc
89.248.165.14	69	202425	Amsterdam/Noord-Holland	Netherlands	FiberXpress BV
42.5.68.19	60	265691	Comas/Lima	Peru	Wi-Net Telecom S.A.C

Extracting the data, we can see where the majority of the IP addresses originate from. Five of the ten Top attacker IP addresses originate in the US in either New Jersey or Michigan. We can also see an example of an ISP spanning multiple geographical areas, as ASN 135377 has geolocations both in Hong Kong and London. This could be through spoofing or they have a server running in both locations.

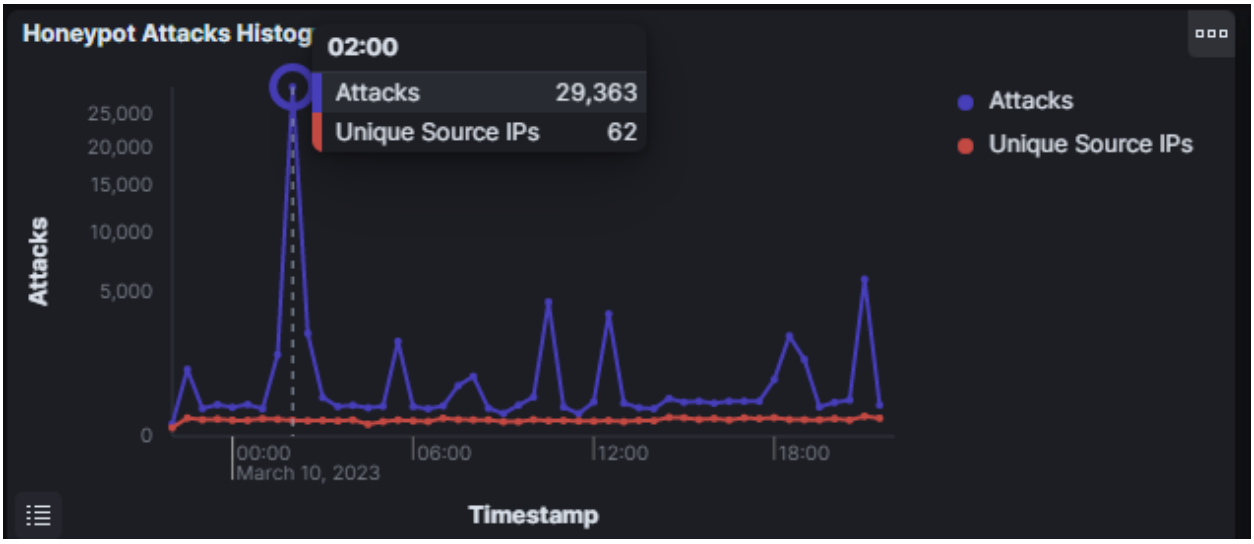
Discovering and hiding NMAP Scans

After I setup my AWS Instance, I decided to run an NMAP port scan on it, and decided to use -A and -sV scan options. The -A flag enables OS and version detection, while also executing built-in scripts for further enumeration while -sV flag attempted to determine the version of the services running at that address. I then went to sleep, but when I woke up, I had way more honeytrap attacks then I was expecting. It appears that during the 6 hour window between March 9th 2200 and March 10th 0400, that nmap -A and -sV scan singlehandedly increased the number of attacks by about 30,000. This really shows when performing reconnaissance, even single port scan commands can set off red flags and alarms everywhere.

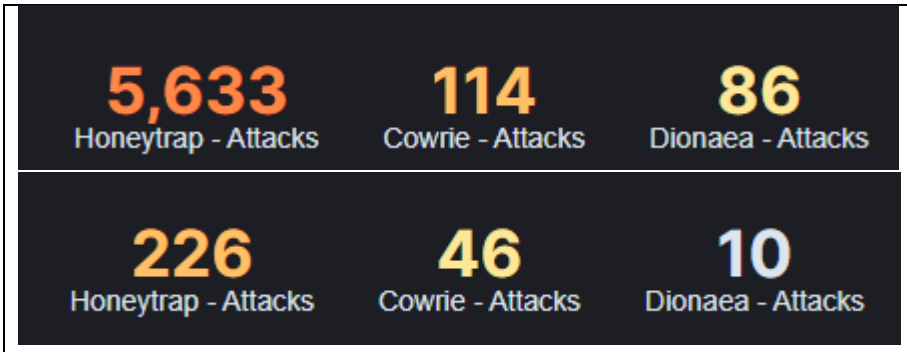
vs

I know it was the result of the NMAP scan because my scan started at 02:04 EST, and the massive spike of attacks occurred between 02:00 EST and 02:30 EST as well.

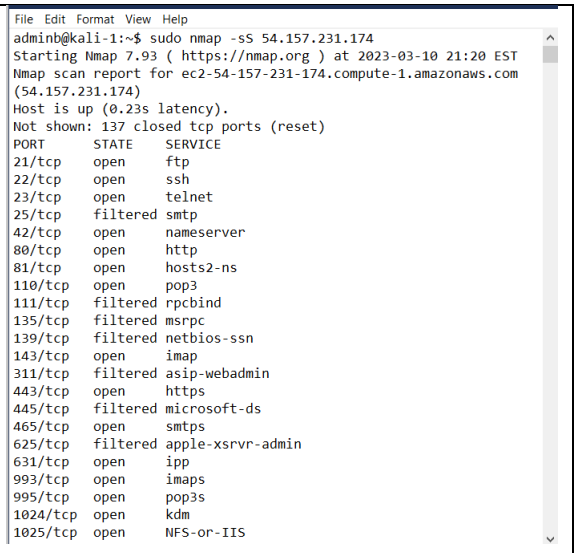
```
adminb@kali-1:~$ nmap -A 54.157.231.174
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-10 02:04 EST
Stats: 0:00:01 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 1.75% done; ETC: 02:06 (0:01:52 remaining)
Stats: 0:00:03 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 43.83% done; ETC: 02:04 (0:00:04 remaining)
Stats: 0:00:09 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 0.59% done
Stats: 0:00:33 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 1.41% done; ETC: 02:38 (0:32:38 remaining)
Stats: 8:23:08 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.37% done; ETC: 10:31 (0:03:04 remaining)
Stats: 8:23:09 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.37% done; ETC: 10:31 (0:03:03 remaining)
Stats: 8:23:09 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.37% done; ETC: 10:31 (0:03:03 remaining)
Stats: 8:23:10 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.37% done; ETC: 10:31 (0:03:02 remaining)
Stats: 8:23:12 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.37% done; ETC: 10:31 (0:03:02 remaining)
Nmap scan report for ec2-54-157-231-174.compute-1.amazonaws.com (54.157.231.174)
Host is up (0.10s latency).
Not shown: 149 filtered tcp ports (no-response)
Bug in ganglia-info: no string output.
Bug in ganglia-info: no string output.
PORT      STATE SERVICE               VERSION
21/tcp    open  ftp                   vsftpd 2.0.8 or later
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_Can't get directory listing: PASV IP 172.28.0.2 is not the same as 54.157.231.174
22/tcp    open  ssh                   OpenSSH 7.9p1 (protocol 2.0)
| ssh-hostkey:
|   1024 d7a5f213586463ab0ce98b42a7d58cdf (DSA)
|   2048 2e7680fb324745d5ecacb7d6291da432 (RSA)
|   256 20c2ec43733729382499938d518f17ee (ECDSA)
|_  256 800eb4e15fa05d27a25297b58dc5d4d2 (ED25519)
23/tcp    open  telnet?
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP, JavaRMI, LANDesk-RC, LDAPBindReq, NULL, NotesRPC, TerminalServer, WMSRequest, X11Probe, afp, giop, tn3270:
|     login:
|   FourOhFourRequest, GenericLines, HTTPOptions, LDAPSearchReq, RTSPRequest:
|     login:
|     Password:
|     Login incorrect
|     login:
|   GetRequest:
|     login:
|     Password:
|     programs included with the Debian GNU/Linux system are free software;
|     exact distribution terms for each program are described in the
|     individual files in /usr/share/doc/*/copyright.
|     Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
|     permitted by applicable law.
|     [4hGET / HTTP/1.0@ubuntu:~$
|   Help, Kerberos, LPDString, SSLSessionReq, TerminalServerCookie:
|     login:
|     Password:
```



While these commands were able to give me detailed information about my AWS instance, they also generated way too much traffic, and I had to exclude my IP Address entirely to not skew the data. Later that day I decided to perform another experiment, trying NMAP but this time with the stealth mode, or the -sS flag. The -sS option is used to perform a TCP SYN scan, also known as a stealth scan. This type of scan sends a SYN packet to the target system's ports, and if the port is open, the target system responds with a SYN-ACK packet. The -sS scan is considered a stealthy scan because it tries to avoid detection by the target system's firewall and intrusion detection system (IDS). The results were immediate, during the same time period, my scan had only generated about 5,000 attacks, only a 1/6th of how many the verbose scans from earlier did.



Even performing the same command, using stealth mode or being aware of how much network traffic you're generating can make your surveillance infinitely less detectable and noticeable if you're a red teamer. The inverse applies too, because if you see a massive increase of attacks in a short period of time from a single IP, you can probably infer that someone is scanning your system for vulnerabilities or open ports.



Problems Encountered:

I did encounter a few problems, mainly with extracting the data from the Elasticvue servers. They are very finnicky with the wording of requests. Also recording false positives, like in the case of my NMAP scan. While it was ultimately useful in determining the effect of Nmap scans on networks, it still can skew the data if left unnoticed. Also this server costs quite a bit, as 18 cents per hour adds up to 4$ a day, and since I have two servers running on opposite sides of the country, it quickly adds up. So I'll likely run this for a week and then make a decision on whether to continue further.

Conclusions and Trends:

The analysis of the Dionaea, Cowrie, and Honeytrap honeypots provides valuable insights into the behavior of cyber attackers and the vulnerabilities they exploit. Dionaea recorded 17,764 attacks from 145 unique IPs, with attackers primarily targeting port 445 and using the SMBD protocol. These attacks originated mainly from Mexico, Central America, Egypt, India, and Taiwan, and peaked at certain times. Meanwhile, Honeytrap had 6979 attacks from 1081 unique IPs, with port 8088 being the most targeted. The majority of attacks on Honeytrap were domestic, with the US being the top source of attacks, followed by Hong Kong, Canada, the Netherlands, and China. Notably, an NMAP port scan on the AWS Instance increased the number of Honeytrap attacks by about 30,000, so maintaining a low profile while conducting reconnaissance is crucial in avoiding detection.

In terms of vulnerabilities, the most frequently exploited vulnerabilities include CVE-2001-0540, CVE-2012-0152, CVE-2002-0013, CVE-1999-0265, and CVE-2019-11500, targeting mainly Windows, Oracle, and Drupal servers. The most prevalent attack signatures were the DoublePulsar Backdoor/EternalBlue, which exploits a vulnerability in Microsoft Windows systems to install a backdoor on the target system, allowing persistent access and control. Another significant attack is the Nmap port scanning activity used with a TCP SYN scan and window size of 1024. Finally CVE-2020-11899 Multicast out-of-bound read exploit is used to execute arbitrary code by exploiting a vulnerability in Windows Hyper-V. The aforementioned attacks, though exploiting mostly patched CVEs, still pose a significant threat to network security and are frequently used by threat actors to gain unauthorized access.

The timing and location of attacks also provide valuable insights. Attempted Administrator Privilege attacks spike during non-business hours, particularly around 1-2 am, 3 am, and 9 pm. The majority of attacks come from the Middle East, Southeast Asia, and the Eastern United States, with China and Iran being the top attacking countries.

The Cowrie honeypot, while not the most targeted in this study, I still think revealed the most about attacker behavior and motive, and recorded 2840 attacks originating from 192 unique IP addresses and associated with 12 unique HASSH fingerprints. Most of these attacks also originated from the Middle East, Southeast Asia, and the Eastern United States, with China and Iran identified as the top attacking countries. As expected, the more unsecure Telnet protocol was used more frequently than SSH to attempt unauthorized access, and SSH-2.0-Go was the most used SSH client during these attacks. Additionally, it is important to implement strong authentication protocols to mitigate the risk of brute force or dictionary attacks that may exploit commonly used or predictable usernames and passwords. Failure to take these proactive measures could result in the exploitation of vulnerabilities by malicious actors, potentially leading to the unauthorized download of ransomware onto open servers or machines. As evidenced by the significant number of ransomware scripts detected and captured by the Cowrie honeypot, the consequences of leaving servers vulnerable to unauthorized access can be severe and costly.

In the next few days of running the honeypot, it is likely that attackers will continue to attempt to gain unauthorized access through Telnet and SSH protocols, using unique SSH clients and HASSH fingerprinting to evade detection. Looking to the future, network trends suggest that attackers will continue to target common ports used by web applications and proxy servers, with certain geographic regions remaining high-value targets. The continued use of Telnet connections, despite their inherent insecurity, is also expected. Organizations must remain vigilant and implement strong authentication protocols, intrusion detection and prevention systems, and ongoing monitoring and analysis of network traffic to prevent unauthorized access attempts and mitigate potential cyber threats.

Overall, the data and insights provided by the analysis of honeypot attacks show the importance of network security best practices, including the use of strong passwords, ongoing monitoring and analysis, and proactive measures to prevent unauthorized access attempts. By implementing these measures, organizations can enhance the resilience of their systems and protect their sensitive data against cyber-attacks.