Honey Pot Analysis Report

Observation data from a cloud-hosted T-Pot instance and the impact of implemented countermeasures.

Overview

A honeypot is a security mechanism that simulates a vulnerable system to attract and monitor potential attackers. It looks like a real part of a network or system but is isolated and monitored to record attack behaviour without risking actual data or operations.

T-Pot is a sophisticated and popular honeypot framework that easily deploys multiple honeypot systems via docker. It utilizes the Elastic stack to provide data visualization and insights into how attackers interact with the honeypots.

Methodologies

Deploying honeypots in any network is risky, as it requires many, if not all ports to be open for inbound traffic. Therefore, I decided to host the machine using cloud services. Microsoft Azure gives sufficient flexibility to easily set up T-Pot and its services. The D series V3 is a good cost-effective option to create VMs with efficient processing power for smooth T-Pot operation. The D4s_v3 with 4 virtual CPU cores and 16GB of RAM provided minimal power to run the T-pot smoothly without crashing.

Their server is located in the East of the US. The experiment is run from Nov 06th 2024

Tpot results

Data points

- At the beginning of the experiment, Tpo receives on average 60,000 attacks a day. However, after 2 weeks, the number grew to more than 100,000.
- On the day of Christmas, the machine received the highest number attack in a day of 42k attacks.
- The single most attempted username is “root” across all services, attempted twice than the second most common username “admin”. Common passwords include “admin”, no password, and “root”. To counter this, simply disabling or renaming the root or admin user could be an effective way to mitigate the majority of credential-based attacks
- Almost half of the total attacks originated from China and the majority of the attempts focused on port 25, which is commonly used by the SMTP service. This could be an

indication that the attackers are interested in intercepting emails for data exfiltration or phishing.
- The US hosts around a quarter of the total attacks, with the most common port being 123, which is commonly used by the NTP service. Although there are potential vulnerabilities where NTP can be manipulated, attackers are likely using the NTP service to perform amplified DDoS attacks on another system they are attacking.
- Although the nature of the attackers varies. Over 90% of traffic originates from known attackers, and most of them use personal or corporation ISPs as opposed to cloud providers. This suggests that filtering traffic using a known attacker IP database could be an effective way to prevent attacks.

Top performing containers

Some containers detected more attacks than others. Below is a summary of the data collected over 15 days.

Cowrite (845,459 attacks)

Corwrite Honey Pot detected approximately 90% of the total attacks. Cowrite is an SSH and Telnet honeypot that logs both the credentials attempted by attackers and the commands they try to execute. Here are some of the most frequently used commands:

Command	**Count**	**Command**	**Count**
enable	34,320	ping ;sh	17,063
shell	34,320	/bin/busybox cat /proc/self/exe || cat /bin/echo	17,025
system	34,318	top	4,552
sh	17,160	lscpu | grep Model	4,544
cat /proc/self/exe	17,063	cd ~; chattr -ia .ssh; lockr -ia .ssh	2,455

Dionaea (36,206 attacks)

Dionaea is a low-interaction honeypot designed to emulate vulnerable network services to capture and analyze malicious activities. Here are the most commonly accessed protocol

protocol	Count	protocol	Count
smbd	18,842	mysqld	577
ftpdatalisten	10,616	mongod	551
mssqld	2,419	pptpd	130
ftpd	2,079	epmapper	127
httpd	686	mqttd	77

When hosting those commonly probed services, on top of proper configuration and hardening, consider changing the port the service is hosted on too.

Sentrypeer (26,105 attacks)

SentryPeer is a honeypot designed specifically to monitor and analyze SIP (Session Initiation Protocol) traffic. It is commonly used to detect and study unauthorized or malicious activities targeting VoIP (Voice over IP) systems.

There are a few reasons an attacker might be interested in exploiting SIPs

- **Toll fraud**: Unauthorized calls to premium numbers for financial gain.
- **Eavesdropping**: Intercepting voice calls or sensitive information.
- **Credential theft**: Stealing SIP account credentials for malicious use.
- **Denial of Service (DoS)**: Overwhelming SIP servers to disrupt services.

Countermeasures

Snort

Snort is a popular open-source Intrusion Prevention System capable of operating in either IDS or IPS mode. It uses a ruleset and a blocklist to analyze traffic, either by signature or IP.

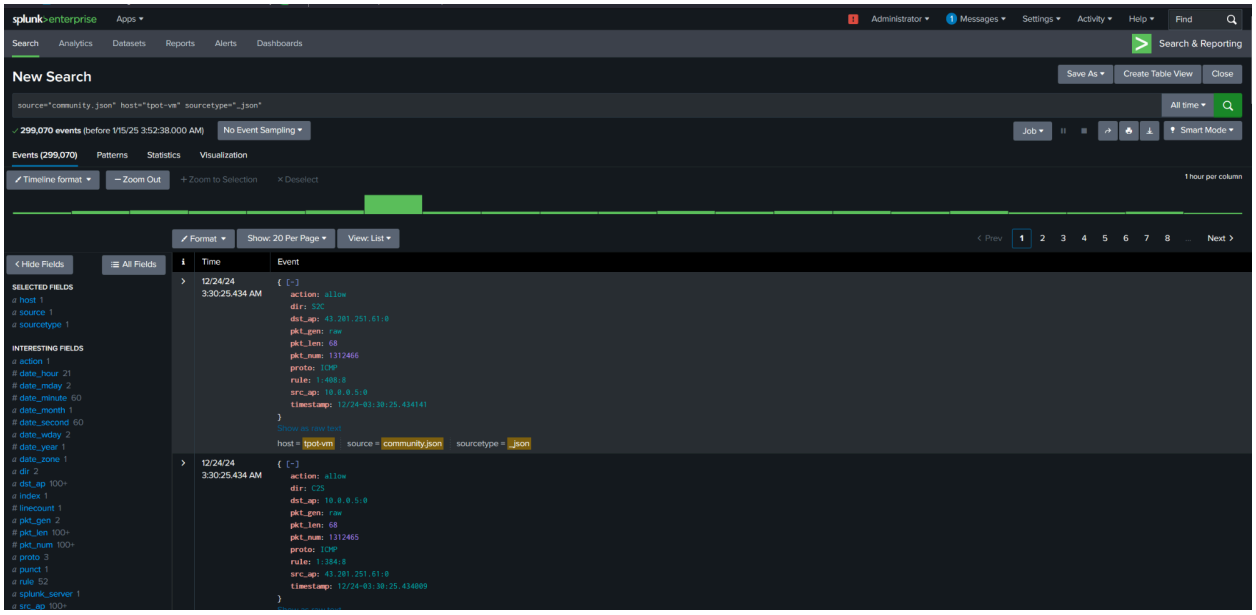
In addition to creating rules and blocklists manually, the PulledPork tool offers an easy way to download commonly used resources. For rulesets, the Community, Snort, and LightSPD versions are available. For blocklists, there are the Snort and LightSPD blocklists.

For this experiment, each snort runs each ruleset for around 24 hours. An analysis of the result is shown below.

Ruleset/Blocklist name	Notes
Snort ruleset (9887 rules)	The Snort ruleset mainly focuses on home personal computer use on the Windows platform, targeting browser exploits and trojan protection, which leads to fewer triggers in this experiment. The two significant events that triggered most frequently were: ● PHP remote code execution attempts ● Windows SMB remote code execution.
Community ruleset (4017 rules)	This ruleset addresses most exploit cases. There were many ICMP-related triggers, which might be disabled to reduce excessive alerts. Other than that, some interesting events were detected: ● Unusual VOIP INVITE messages, FTP login attempts, SNMP requests, and potential malware activities. ● Specific suspicious traffic such as DoublePulsar variant injections, SMB anonymous access, and Microsoft cmd.exe banner. ● Multiple protocol-related events, such as DNS spoofing, TFTP requests, and SQL ping attempts.
LightSPD ruleset (9887 rules)	This ruleset only got triggered 7 times. But each event is very specific, such as: ● Sonatype Nexus Repository Manager remote code execution attempt ● TP-Link Archer Router command injection attempt ● The Mirai and Remcos malware
LightSPD blocklist (1390 IPs)	It blocked 103.102.230.0 179 times
Snort blocklist (1574 IPs)	Nothing was triggered in this list, evidently, both blocklist doesn't perform well

Splunk

Splunk is a powerful software platform used for searching, monitoring, and analyzing machine-generated big data via a web-based interface. It collects and indexes data from various sources, including servers, applications, and network devices, making it searchable in real time. Snort reports can be better presented and iterated on using splunk, and it also has built in sensors for simple web traffic.



Process load

Both Tpot and Snort doesn't generate a significant amount of traffic, a 128GB hard drive should provide efficient storage

The cloud machine selected is equipped with Standard D4s v3 (4 vcpus, 16 GiB memory). With the top output shown below, it remains smoothly responsive after running Tpot and snorts concurrently.

```
top - 08:59:33 up  5:28,  1 user,  load average: 0.59, 0.69, 0.81
Tasks: 333 total,   1 running, 332 sleeping,   0 stopped,   0 zombie
%Cpu(s):  4.6 us,  1.5 sy,  0.0 ni, 92.7 id,  0.8 wa,  0.0 hi,  0.3 si,  0.0 st
MiB Mem :  15954.6 total,   3807.6 free,   8591.7 used,   3896.9 buff/cache
MiB Swap:      0.0 total,      0.0 free,      0.0 used.   7362.9 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   6255 tpot      20   0 6559228   2.8g  63148 S  10.3  17.7  35:27.83 java
   6826 tpot      20   0   21.1g 516660  53320 S   4.3   3.2  20:09.01 node
   6744 tpot      20   0 5099856   1.6g  60172 S   3.7  10.4  23:59.86 java
 117687 root      20   0  216940 134256  13432 S   1.7   0.8   1:40.37 snort
    623 root      20   0 4824876 128148  56436 S   1.0   0.8   2:49.59 dockerd
   2706 tpot      20   0  562492 533760  52112 S   0.7   3.3   4:23.93 Suricata-Main
   5421 tpot      20   0   45348  35476   9332 S   0.7   0.2   1:08.05 python3
   6818 tpot      20   0   48336  41928   9552 S   0.7   0.3   1:24.80 python3
    608 root      20   0 2466984  67656  33404 S   0.3   0.4   0:47.94 containerd
    952 root      20   0  330848  33088  11592 S   0.3   0.2   0:20.80 python3
```

Installation pain points

Tpot (Commit f6398f9)

1. Make sure the path where the Tpot is cloned is large enough. Around 128GB should be sufficient.
2. There's currently a bug where the "WEB_USER" field isn't filled automatically in the .env file. Make sure to fill the with the base64 encoded string of your web username.
3. There will be many conflicting ports. Use lsof to detect the service and terminate them.

Pulled pork (v3.0.0.5)

1. There's currently a bug where the snort blocklist link leads to a terms and service page instead of the actual blocklist. Make sure to manually download it
2. There's currently a bug where the "registered_ruleset" version is broken, you can fix it by overwriting this in the pulledport.py and hardcoding "SNORT_BLOCKLIST_URL"
3. The lightSPD ruleset is somehow being recognized as a virus by windows defender

Sources

T-Pot Repository
The official T-Pot repository can be found on GitHub:
[T-Pot GitHub Repository]

Hosting a Honeypot in Azure Cloud using T-Pot
For a detailed guide on hosting a honeypot in Azure using T-Pot, refer to the following article:
[Hosting a Honeypot in Azure Cloud using T-Pot]