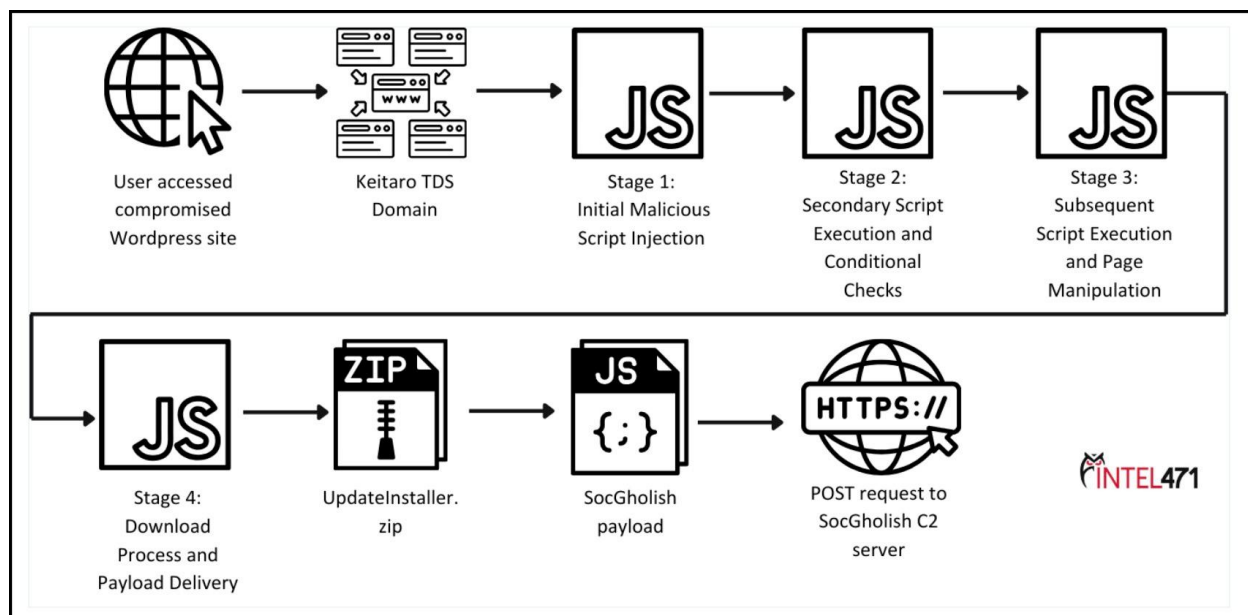


Threat hunting case study: SocGholish

 intel471.com/blog/threat-hunting-case-study-socgholish

SocGholish is an ongoing malware campaign that started around 2017 and involves the distribution of malware via hacked websites. These trusted sites, which surface organically in search results, are selected so as to not raise suspicions and are used to trick users into downloading and installing malware disguised as legitimate software. SocGholish attackers have also been known to use domain shadowing, which involves compromising domain registrar accounts and creating subdomains that they then incorporate into their attack chains. Again, this technique piggybacks on the trust in the main domain.

The SocGholish malware often poses as a browser update. When users visit sites that have been compromised, they are prompted to install the update. A recent campaign warned a victim that they were using an older version of Microsoft's Edge browser. The malware is delivered using a convoluted infection chain designed to thwart security checks that might otherwise stop the malware. This can involve JavaScript, PowerShell and the use of compressed (ZIP) files. Delivery typically requires user interaction, such as clicking on a file. The attacker can then move to install a secondary payload. In the past, those payloads have ranged from WastedLocker ransomware to Cobalt Strike, the post-exploitation framework used by both attackers and red teams.



An illustration of the steps that lead to a SocGholish infection.

As part of distributing SocGholish at scale, its operators have used a sophisticated traffic distribution and redirecting system called Keitaro. It is intended for advertisers to optimize campaigns by being able to funnel users to different kinds of content based on their location or browser type or device. Advertisers do this to create more conversions. Malware

operators use it for the same purpose: increasing the likelihood of successful infections while attempting to mask campaigns from threat researchers. This sophisticated approach to victim selection showcases a strategic effort to optimize impact and financial returns. By concentrating on targets that are more likely to pay ransom or where sensitive data can be exploited, SocGholish's operators exhibit a deep understanding of the value of their potential victims.

SocGholish infections have been primarily linked to the threat actor group **Evil Corp** aka **Mustard Tempest**, **DEV-0206**, **TA569**, **UNC1543**. Members of this organized crime group have been sanctioned by the U.S. in 2019 and by the U.S., Australia and the U.K. in 2024 while other alleged members have been indicted (Maksim Yakubets and Aleksandr Ryzhenkov). The group has long-running cybercrime links going back to the Zeus and Dridex banking malware programs and has been involved in money laundering and ransomware. The SocGholish campaigns are designed to gain initial access to a steady supply of computers that can then be monetized in other ways, whether through data theft from those machines or ransomware. The campaigns have a significant reach. In one example, we identified a SocGholish campaign from October 2024 that generated more than 1.5 million interactions in a one-week period.

We can threat hunt to uncover signs of possible SocGholish infections and remove malware before systems are exploited further. Intel 471 has written several hunt packages for our HUNTER471 threat hunting platform, which can help security teams proactively detect possible malicious behavior. What follows is an example of how a technique commonly used by SocGholish operators can be detected.

Task Scheduler (formerly known as Scheduled Tasks) is a native Windows binary that allows administrators to schedule various actions when certain conditions have been met, such as starting an application after the system has been rebooted. Attackers abuse schtasks.exe often to ensure their malware executes or is persistent, and it is one of the techniques listed in MITRE ATT&CK — the catalog of attacker tactics, techniques and procedures (TTPs). Many malware campaigns and their accompanying threat groups use this technique, including SocGholish, GootLoader (see: "Threat hunting case study: Tracking down GootLoader"), Spectre remote access tool (RAT), NetSupport RAT, Lumma information-stealing (infostealer) malware and the ransomware groups **MedusaLocker**, **Nokoyawa**, **Quantum** and **LockBit 3.0**.

Rather than allow a malicious scheduled task to execute from the normal Windows directory, attackers often place it elsewhere in the file system to make it more difficult to find. Uncovering a scheduled task from an abnormal location can be a clue to a possible malware infection, and we can conduct a structured hunt for this behavior.

First, we can often find evidence of a new scheduled task in Windows event logs if that level of auditing is enabled. Event code 4698 captures this and lists the task name, command used and task arguments, which are the most interesting artifacts that can inform the threat hunt. Another avenue with which to find malicious scheduled tasks is through the process-create style event codes such as 4688, which is the native in Windows event logging code for creating a new process. System Monitor (Sysmon) users will capture the same activity, and endpoint, detection and response (EDR) security solutions may also record this activity. Additionally, adversaries may create a scheduled task through the command line or a scripting interpreter such as PowerShell, offering another hunting opportunity to detect those commands an adversary ran. Now, let's go to the hunt package in HUNTER471.

The screenshot displays the HUNTER471 web interface for a specific hunt package. The top navigation bar shows the path: Research > Search > Hunt Package > Scheduled Task Executing from Abnormal Location. The main header area includes a 'HUNT PACKAGE' button, the title 'SCHEDULED TASK EXECUTING FROM ABNORMAL LOCATION', a star icon, and a 'FEEDBACK' button. The left sidebar contains a list of navigation items: My Environment, Detection Rules, Overview, Description, Emulation and Validation, Query Logic, Research, Analyst Notes, Threat Description, References, MITRE Information, and Response Actions. The main content area is divided into sections: 'MY ENVIRONMENT', 'DETECTION RULES' (with 'STAGE FOR DEPLOY' and 'MARK AS DEPLOYED' buttons), and 'OVERVIEW'. The 'OVERVIEW' section includes a 'DESCRIPTION' and a table with columns: CREATED, LAST UPDATED, SEVERITY, and THREAT NAMES.

CREATED	LAST UPDATED	SEVERITY	THREAT NAMES
May 4, 2022	Jan 29, 2025	High	MEDUSALOCKER, NOKOYAWA,
			QUANTUM RANSOMWARE,
			LOCKBIT 3.0, NETSUPPORT,
			GOOTLOADER,
			LUMMA INFESTEALER,
			SOCGHOLISH, SPECTRE RAT

A HUNTER471 hunt package that looks for scheduled tasks executing from abnormal locations.

Scrolling down, we can see a description of the hunt package:

This hunt package is designed to capture activity associated with a scheduled task which includes abnormal locations in its details for execution. This is often a mark of persistence or malicious tasks created by malware or attackers.

Persistence is one of the reasons attackers abuse scheduled tasks, but there are other methods used. Sometimes, attackers will create a scheduled task, use it to run arbitrary code and then delete itself. This behavior could indicate the second or third stage of an attack. These are the type of aspects that should be considered when looking at artifacts of living-off-the-land binaries, which are native Windows tools that are used by attackers to avoid detection.

The query logic table of this threat hunt as seen below broadly explains its aim. In this case, we can see a series of values. These locations may not necessarily be abnormal for your environment but are good starting points. Often, these locations come from past intelligence and data forensics reports describing adversary activity. The Users\Public\ directory is constantly abused by adversaries as well as \Perflogs\, which is short for Performance Logs, which are log files stored by Windows. "ProgramData" is used to store application data.

QUERY LOGIC ⓘ		
Selection	Field	Value
ScheduledTask	event_id	4698
ScheduledTask (ANY)	message	"Users"
	message	"Public"
	message	"Perflogs"
	message	"\$Recycle"
	message	"ProgramData"
keywords (ANY)	keywords	\Windows Defender\
	keywords	\Microsoft\Windows\Application
Condition: ScheduledTask and not keywords		

The query logic for uncovering scheduled tasks in abnormal locations.

Let's start hunting. This threat package covers numerous EDR and security incident and event management (SIEM) systems: CarbonBlack Cloud - Investigate, CarbonBlack Response, CrowdStrike, CrowdStrike LogScale, Elastic, Microsoft Defender, Microsoft Sentinel, Palo Alto Cortex XDR, QRadar Query SentinelOne, Splunk, Tanium, Tanium Signal and Trend Micro Vision One.

We will conduct two different hunts, with the first using Sysmon logs. This query looks for scheduled tasks executing from abnormal locations using command-line arguments.

New Search

```

1 | index=sysmon ((Image=="schtasks.exe" OR CommandLine IN ("*\ats*" "schtasks*")) (CommandLine IN ("*Users*" "*Public*" "*Perflogs*" "*$Recycle*" "*ProgramData*" NOT (CommandLine IN ("*\Windows Defender\*" "A:\Microsoft\Windows\Application" "A:\Microsoft\ClickToRun*"))))
2 | rename Computer as hostname CommandLine as processCmdLine Image as process process_id as processId ParentImage as parentProcess parent_process_id as parentProcessId user as username
3 | stats min(_time) as firstSeen max(_time) as lastSeen values(username) as username values(process) as process values(processId) as processId values(parentProcess) as parentProcess values(parentProcessId) as parentProcessId values(processCmdLine) as processCmdLine by hostname
4 | convert ctime(*Seen)

```

7 events (1/14/25 5:00:00.000 PM to 1/21/25 5:07:25.000 PM) No Event Sampling

hostname	firstSeen	lastSeen	username	process	processId	parentProcess	parentProcessId	processCmdLine
DESKTOP-28BWW1B.lexicorp.local	01/17/2025 16:08:47	01/17/2025 16:08:47	NT AUTHORITY\SYSTEM	C:\Windows\System32\schtasks.exe	6316	C:\Windows\System32\cmd.exe	12288	schtasks /CREA
Desktop-Tan1.lexicorp.local	01/17/2025 11:57:01	01/17/2025 11:57:01	LEXICORP\jamesmurphy	C:\Windows\System32\cmd.exe	2488	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	2488	"C:\Windows\sys
				C:\Windows\System32\schtasks.exe	9140	C:\Windows\System32\cmd.exe	384	/c calc.exe";n
								ActiveXObject("
								> C:\Users\P
								"Cyborg_Test"
								schtasks /crea
Win10-Dirty-Goo.lexicorp.local	01/17/2025 13:10:32	01/17/2025 13:10:32	LEXICORP\jamesmurphy	C:\Windows\System32\cmd.exe	5252	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	4424	"C:\Windows\sys
				C:\Windows\System32\schtasks.exe	6648	C:\Windows\System32\cmd.exe	6648	/c calc.exe";n
								ActiveXObject("
								> C:\Users\P
								"Cyborg_Tes

A threat hunt query using Sysmon logs looking for scheduled tasks executed from abnormal locations using command-line arguments.

In the results, we can see that a username exists, and we see the attacker had system level privileges and ran “schtasks.exe.” We can also see the parent process was “command.exe.” Scrolling all the way to the right, we can see the process command line contains the artifacts we’re looking for.

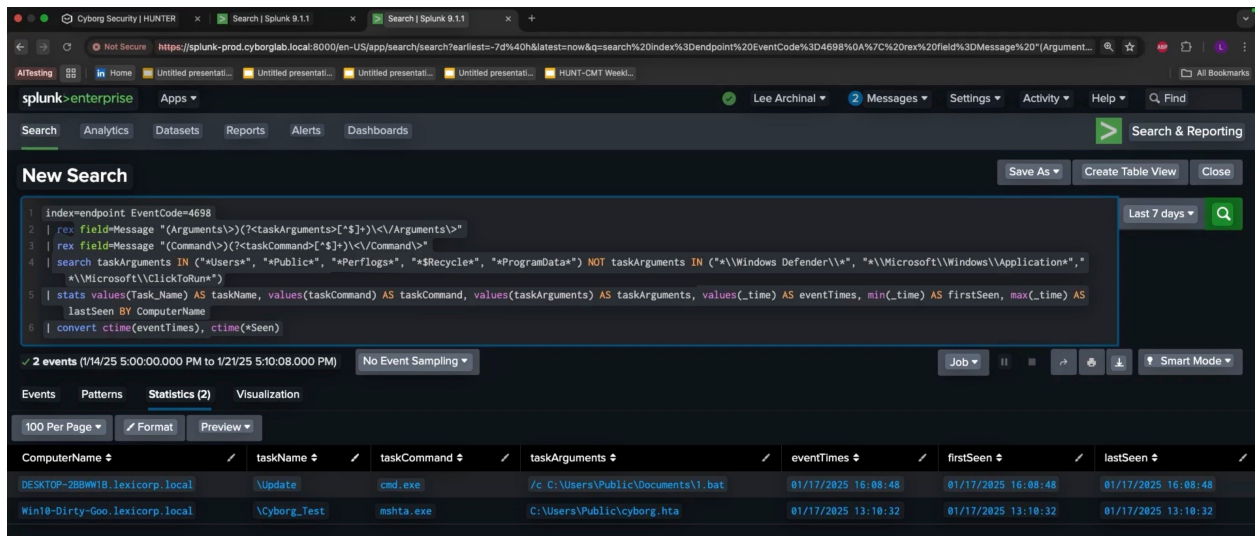
7 events (1/14/25 5:00:00.000 PM to 1/21/25 5:07:25.000 PM) No Event Sampling

parentProcessId	processCmdLine
12288	schtasks /CREATE /TN "Update" /TR "cmd.exe /c C:\Users\Public\Documents\1.bat" /SC MINUTE /MO 10 /RL HIGHEST /F
2488	"C:\Windows\system32\cmd.exe" /c "echo <html><head><HTA:APPLICATION ID="AtomicTest"><script language="jscript">var
384	/c calc.exe";new
	ActiveXObject("WScript.Shell").Run(c);</script></head><body><script>self.close();</script></body>"
	> C:\Users\Public\cyborg.hta & schtasks /create /tn "Cyborg_Test" /sc onlogon /tr "mshta.exe C:\Users\Public\cyborg.hta" & schtasks /run
	"Cyborg_Test"
	schtasks /create /tn "Cyborg_Test" /sc onlogon /tr "mshta.exe C:\Users\Public\cyborg.hta"
4424	"C:\Windows\system32\cmd.exe" /c "echo <html><head><HTA:APPLICATION ID="AtomicTest"><script language="jscript">var
6648	/c calc.exe";new
	ActiveXObject("WScript.Shell").Run(c);</script></head><body><script>self.close();</script></body>"
	> C:\Users\Public\cyborg.hta & schtasks /create /tn "Cyborg_Test" /sc onlogon /tr "mshta.exe C:\Users\Public\cyborg.hta" & schtasks /run
	"Cyborg_Test"
	schtasks /create /tn "Cyborg_Test" /sc onlogon /tr "mshta.exe C:\Users\Public\cyborg.hta"

A screenshot showing the command-line arguments related to a scheduled task event in Windows.

We see the task name, TN, is “Update.” The task requirements point to the user’s public directory and a bat file. That file is scheduled to run every 10 minutes at the highest privileges. It’s unclear at this point if the bat file is malicious, but since it is meeting the parameters of odd behavior, it’s worth investigating further.

We can also hunt another way using the scheduled task telemetry. If those command-line arguments above were successful, we should also be able to tie those to event code 4698. In this hunt, we'll switch to running a command-line telemetry query in Splunk.



```
1 | index=endpoint EventCode=4698
2 | rex field=Message "(Arguments>)(?<taskArguments>[*$]+)<\/Arguments>)"
3 | rex field=Message "(Command>)(?<taskCommand>[*$]+)<\/Command>"
4 | search taskArguments IN ("*Users*", "*Public*", "*Perflogs*", "*$Recycle*", "*ProgramData*") NOT taskArguments IN ("*\\Windows Defender\\*", "*\\Microsoft\\Windows\\Application*", "*\\Microsoft\\ClickToRun*")
5 | stats values(Task_Name) AS taskName, values(taskCommand) AS taskCommand, values(taskArguments) AS taskArguments, values(_time) AS eventTimes, min(_time) AS firstSeen, max(_time) AS lastSeen BY ComputerName
6 | convert ctime(eventTimes), ctime(*Seen)
```

ComputerName	taskName	taskCommand	taskArguments	eventTimes	firstSeen	lastSeen
DESKTOP-288W18.lexicorp.local	\Update	cmd.exe	/c C:\Users\Public\Documents\1.bat	01/17/2025 16:08:48	01/17/2025 16:08:48	01/17/2025 16:08:48
Win10-Dirty-Goo.lexicorp.local	\Cyborg_Test	mshta.exe	C:\Users\Public\cyborg.hta	01/17/2025 13:10:32	01/17/2025 13:10:32	01/17/2025 13:10:32

A query hunting for certain command-line events in Splunk.

Under the heading “New Search” is the query, which looks for event code 4698. The query is written to extract the events of interest from the message field, including taskName, taskCommand and taskArguments. We can see the results are the same as the first query, showing a scheduled task involving the “1.bat” file. Whether you start from either query, you'll find the same artifacts if the command-line argument was successful and the scheduled task was created.

We hope this tutorial has been useful. A video version is available [here](#). For more threat hunting content related to the **Evil Corp** group, see our blog "[Threat hunting case study: Looking for Evil Corp.](#)" Register for a HUNTER471 [Community Account](#), which contains sample free hunt packages along with a comprehensive library of advanced threat hunting packages, detailed analyst notes and proactive recommendations. These resources are designed to strengthen your threat hunting capabilities and keep your organization secure. Happy hunting!