

Eclipse Ransomware – Tradecraft Exposed Through a Leaked C2 server

Author: Piotr Bienias

No of pages: 26

Read time: ~23 minutes

Contents

1	Introduction	3
2	Who is Eclipse?.....	4
3	Victim profile.....	6
4	Toolkit.....	7
4.1	“Streamer” – custom made post-exploitation framework.....	7
4.2	Weaponized Trend Micro Products	9
4.3	DLL-based payload delivery	10
4.4	Custom Keyloggers	10
4.5	Off-the-Shelf Tools	11
5	Infrastructure exploitation	12
5.1	VMware ESXi Compromise.....	12
5.2	Veeam Backup Infrastructure Abuse	12
5.3	Assessment	13
6	Lateral movement and persistence.....	14
6.1	Encryptor Self-Spread.....	14
6.2	Operational Scripts.....	14
6.3	Persistence.....	15
7	Data Exfiltration.....	16
8	The Encryption	17
9	Conclusion	20
10	Indicators of Compromise.....	21
10.1	Network Indicators	21
10.2	Windows Event Log Indicators	21
10.3	EDR / Sysmon Indicators	22
10.4	Active Directory Indicators	23
10.5	Linux / ESXi Indicators.....	23
10.6	File Hashes.....	24

1 Introduction

Most of what the cybersecurity community knows about ransomware groups comes from a scarce number of evidence and artifacts collected after the harm was already done and the group achieved their goal. Bits and pieces collected from various sources over longer periods of time allow to create a profile of the attackers that can later be used for attributing attacks to specific threat actors with a various degree of certainty. It is a very exceptional opportunity to be able to analyze the whole scope of a group's operational framework in one place, and it is a true treasure trove of information that can be used to improve organizations' security postures and better defend against the threats they may pose. This is particularly critical when organizations are facing an active breach by a known threat actor – having access to their complete operational toolkit enables defenders to anticipate the attacker's next moves, identify all implants and footholds, and remediate comprehensively rather than chasing individual indicators.

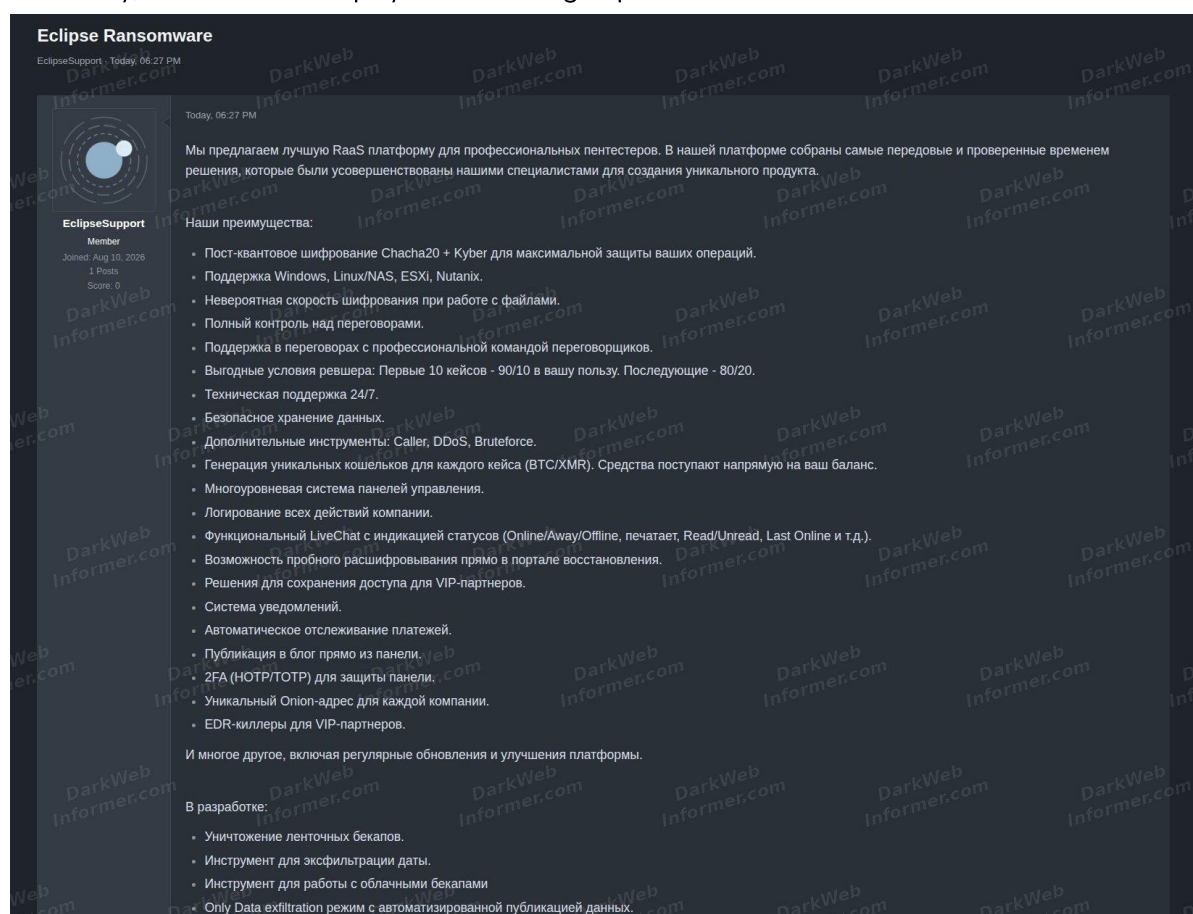
At the beginning of October 2026, **Atos Threat Research Center discovered a C2 server used by the Eclipse Ransomware group** that was exposed to the internet and contained dozens of files comprising an almost **complete operational toolkit**: platform-specific encryptors, exploits for known vulnerabilities, post-exploitation tools, keyloggers, credential harvesting scripts, weaponized Trend Micro binaries, and exfiltration infrastructure targeting at least seven organizations across five countries.

The server that Atos researchers identified was not a disposable C2 that Eclipse used for managing their campaigns but a shared toolbox where they stored everything needed to compromise a network, steal its data, and encrypt it for ransom. What makes this collection uniquely valuable is what it tells us about how the group operates.

The analysis that follows is based on static reverse engineering, without a dynamic one. Its goal is to direct defenders towards specific artifacts and behaviors they should look for in order to identify potential breaches before Eclipse could act on their objectives. TRC collected evidence of active campaigns, operational logs, tooling, and details of the group's techniques, tactics, and procedures. Within Phantom platform, OpenCode functioned as the operator interface, while WormGPT provided the AI assistant, persona, and operational guidance.

2 Who is Eclipse?

Eclipse is a financially motivated double-extortion group that emerged on dark web cybercrime forums in mid-August 2026. Operating under the alias "EclipseSupport", the group launched a Ransomware-as-a-Service (RaaS) program and began recruiting initial access brokers and, offering a centralized administrative console with campaign management, telemetry, and automated payment tracking capabilities.



<https://x.com/DarkWebInformer/status/2087258043826188599/photo/1>

All communications are conducted via the Session messenger and Tox. Negotiations with victims take place through a LiveChat portal hosted on the group's Tor-based infrastructure.

Evidence collected from the C2 server confirm the public profile of the group. All recovered encryptors had the string "Eclipse Ransomware" embedded, version v1.9.2026.01.20, two Tor addresses (negotiation portal and Eclipse Blog) and had a triple locale check that aborts the encryption of 13 CIS-region country code – a well-known mark of operations protected by Russian-speaking jurisdictions. RaaS model of the group is confirmed at the binary level. Two of

the recovered encryptors shared identical code with the only differences being the embedded encryption keys and campaign identifiers allowing for the conclusion that they are an output of an automated builder. Atos TRC also identified two development environments. The encryptors were built on the Linux system with the root account while a custom C2 implant was compiled on a macOS ARM under the username “malcolmbetruger” – a character from a well know game series “Doom”.

3 Victims profile

Analysis of the exposed C2 server identified seven victim organizations across five countries, spanning the education, business process outsourcing (BPO), manufacturing, logistics, and managed services sectors. Educational institutions represented the largest sector, accounting for three of the seven identified victims, while the remaining targets were distributed across commercial organizations in North and South America, Europe, and Asia.

The victim set suggests that Eclipse does not focus on a single industry vertically. Instead, the group appears to prioritize organizations that maintain large volumes of operational, financial, employee, or customer data. Several intrusions involved access to Active Directory databases, credential stores, backup infrastructure, human resources records, student information systems, or cloud administrative assets, indicating an emphasis on compromising systems that provide broad organizational access and valuable data for extortion.

Despite the diversity of sectors, the identified victims share a common characteristic: a relatively limited barrier between initial compromise and access to high-value assets. The recovered tooling and scripts suggest that Eclipse operators routinely pursue credential theft, privilege escalation, and data exfiltration opportunities after gaining access, enabling them to extract sensitive information across a wide range of organizational environments.

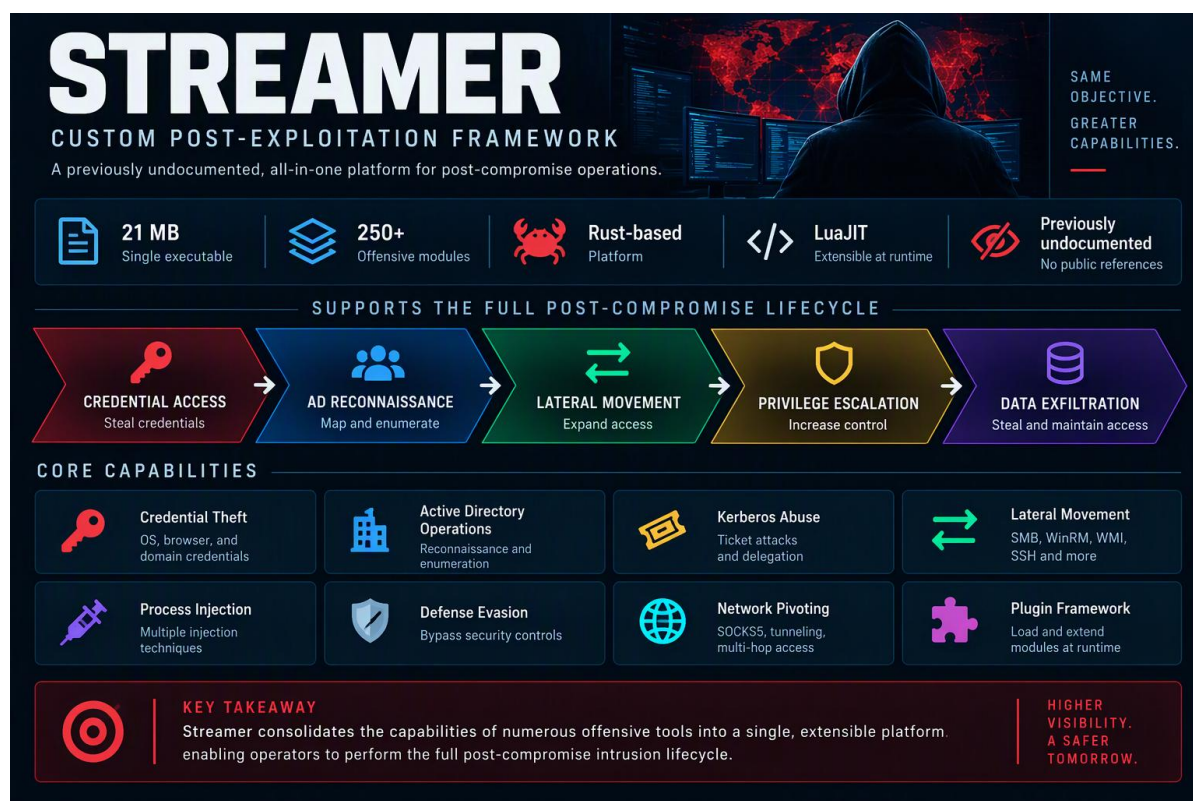
Victim #	Sector	Country	Notable Assets Targeted
1	Call Center / BPO	Chile	Active Directory infrastructure, user credentials
2	K-12 Education	United States	Student records, HR data
3	Auto Parts Manufacturing	Brazil	Payroll, tax, and HR information
4	Logistics	Singapore	Backup infrastructure, domain credentials
5	Higher Education	United States	Domain administrator accounts
6	Managed Services / Google Workspace	Ireland	Cloud administrative assets
7	Higher Education	England	File servers and internal infrastructure

4 Toolkit

Eclipse's arsenal combines custom-built tooling with modified open-source and commercial tools. The custom pieces are what set this operation apart.

4.1 "Streamer" – custom made post-exploitation framework

The most significant artifact on the C2 server is a previously undocumented post-exploitation framework its developer calls "Streamer." The name is not an analyst designation - it was recovered from Rust source paths the compiler embedded into the binary (complex-projects/streamer/), crate names (streamer-lua-ffi), and the LuaJIT build tag (LuaJIT 2.1.0-beta3 (streamer-shim)). No C2 framework by this name appears in public threat intelligence.



"Streamer" – Eclipse custom made post-exploitation framework

Streamer is a 21 MB Rust binary – a single executable that replaces the need for an entire post-exploitation tool suite. It ships with over 250 offensive modules organized into categories the developer labels in the binary itself: keylog, clip, c2, dump, shellcode, implant, exfil, loot, evil, cred, token, passwd, shadow. An embedded LuaJIT scripting engine with seven compatibility shims (JSON, filesystem, network, pattern matching, HTML/XML parsing) allows operators to extend its capabilities at runtime without recompiling.

“Streamer” module inventory spans:

- **Credential theft:** Six LSASS dump techniques (NanoDump, MirrorDump, Lsassy, in-memory, minidump, advanced), SAM and LSA secret extraction, DPAPI decryption, Windows Credential Manager enumeration, Group Policy Preferences password recovery, browser credential theft, and Internal Monologue NTLM hash capture.
- **Active Directory:** Full BloodHound-compatible collection (users, computers, groups, ACLs, trusts, containers, OUs, GPOs), LDAP reconnaissance, AD Certificate Services enumeration, ACL scanning, and vulnerability discovery.
- **Lateral movement:** Native implementations across seven protocols – SMB2/3 (with AES-CMAC signing), WinRM (with embedded SOAP XML templates), WMI via DCOM, Kerberos, DCE/RPC, DRSUAPI, and SSH – plus pass-the-hash, pass-the-key, remote service creation, and remote scheduled task creation.
- **Kerberos attacks:** Kerberoasting, AS-REP roasting, golden and silver ticket forging, pass-the-ticket, ticket injection, and DCSync via DRSUAPI.
- **Process injection:** Five techniques – classic remote thread injection, process hollowing, DLL injection, self-injection, and PE reimaging.
- **Evasion:** AMSI bypass, ETW bypass and patching, DLL unhooking (ntdll refresh from disk), sleep obfuscation with stack/heap encryption, and direct syscalls.
- **SMB reconnaissance:** Over 50 sub-modules covering admin share probing, service enumeration, PetitPotam coercion, PrintNightmare, LAPS checks, WSUS configuration, Sysmon detection, and more.
- **Pivoting:** SOCKS5 proxy, SSH tunneling, and a pivot management system for multi-hop access.
- **Plugin system:** A reflective PE loader allows operators to load, unload, and hot-reload additional modules at runtime.

Twelve functions were decompiled in Ghidra to verify these capabilities were not just source file names. Each proved functional - from the byte-indexed command dispatch jump table, to the CreateProcess/VirtualAllocEx/WriteProcessMemory injection chain, to the CredEnumerateW credential harvesting, to the LDAP search and Kerberos package access routines. The remaining modules are identified from Rust source paths the compiler embedded for panic/debug output. The 21 MB binary size is consistent with 250+ functional modules, but individual modules were not each verified.

The developer metadata embedded in the binary reveals the development environment. Both recovered Streamer builds were cross-compiled on macOS - one on ARM (stable-aarch64-apple-darwin), the other from an Intel macOS host - targeting Windows x64. The username

“malcolmbetruger” appears in the build path (/Users/malcolmbetruger/Documents/Work/code/complex-projects/streamer/), recovered from the Streamer beacon decrypted from the winsvc.exe payload. The C2 management platform behind Streamer is Node.js-based, identified in deployment tokens as nodejs-hypersocks3-v3, communicating over a custom binary protocol called protocol_v3. The protocol's wire format was not reconstructed - that would require dynamic analysis.

Despite its capabilities, Streamer had never been submitted to VirusTotal at the time of analysis. The standalone binary (b7e3a91c.exe) and the beacon recovered from inside winsvc.exe (FBD309CC...) are the only two known builds.

4.2 Weaponized Trend Micro Products

Eclipse repurposes Trend Micro-attributed DLLs as carriers for encrypted payloads. Three distinct TM products were identified through PE version info and PDB paths embedded in the binaries:

Filename(s) on C2 server	OriginalFilename (PE metadata)	TM product	VT detection
winsvc.exe	Corrupted	ATSE64 – core scan engine	23/71
rust.exe, rust_signed.exe, rust.dll, b3f7a1c9.exe	trxHandler (CompanyName: Trend Micro Inc.)	trxHandler – cloud file reputation module	1-2/72
122.exe	TmopsmHttp.dll (CompanyName: Trend Micro, Inc.)	TmopsmHttp – HTTP scan manager	never submitted

Each follows the same pattern: a binary presenting itself as a Trend Micro module carries an 18–21 MB encrypted payload injected into its PE resource section under a randomly generated name (JJKLNBFE4, DXASQTRTGHTU, FUERHEVIUFHBVKGJ). None of the three contain the imports needed to extract their own payloads – an external mechanism is required to load and decrypt the embedded resource. The corrupted version info in winsvc.exe may be a side effect of the payload injection breaking the resource structure.

The higher detection rate of winsvc.exe (ATSE64, 23/71 on VirusTotal, first seen September 2026) compared to the trxHandler variants (1-2/72, first seen October 2026) suggests Eclipse iterated across TM products in response to detection. The latest variant, 122.exe (TmopsmHttp), was never submitted to VT.

Decryption of the winsvc.exe payload was possible because a 32-byte XOR key was found embedded in the binary's .rdata section. The decrypted blob contained a Streamer C2 beacon

compiled on macOS ARM under the malcolmbetruger username. The remaining two variants carry similarly structured encrypted payloads but their decryption keys were not recovered.

4.3 DLL-based payload delivery

Two DLLs recovered from the server — `evil.dll` and `evil2.dll` (both 207 KB) — appear to form part of the delivery mechanism for the weaponized Trend Micro binaries described above. Both are compiled with the same GCC 12/MinGW toolchain as the keyloggers and export printer driver functions (`DrvGetDriverInfo`, `DrvQueryDriverInfo`), suggesting they are designed to be loaded by a legitimate print-related process via DLL sideloading, though the specific host process was not identified on the server.

Analysis of the embedded strings and imports confirms their behavior: each opens a marker file (`pn_marker.txt`) in write mode via `fopen`, writes a status string ("PN loaded as SYSTEM\n" in `evil.dll`, "PN loaded" in `evil2.dll`) via `fwrite`, then calls `WinExec` to execute `cmd.exe /c <path>\pn.cmd`. The marker file serves as a confirmation that the DLL was loaded with the expected privileges. `evil.dll` targets `C:\ProgramData\b8f3a1c2\`, while `evil2.dll` targets a victim's network share — indicating that the same delivery approach was adapted for both local and remote execution scenarios.

The batch script `pn.cmd` was not found on the C2 server and no recovered binary creates it. It is likely deployed separately by the operator and serves as the final link in the execution chain, though its exact contents remain unknown.

4.4 Custom Keyloggers

Two keyloggers (43–44 KB) were compiled 11 minutes apart with the same GCC 12/MinGW toolchain used for the sideloading DLLs — likely the same developer. The first only captures keystrokes when a CDK Global automotive dealer management window is in the foreground, filtering on 11 keywords. The second removes this filter and logs everything. Neither sample contains persistence or exfiltration functionality, indicating that both were intended solely for credential collection, with logs likely retrieved through the C2 infrastructure. Neither has been submitted to VirusTotal.

Notably, the targeted software is widely used throughout the automotive dealership sector. CDK Global itself was the victim of a high-profile ransomware attack executed by Black Suite ransomware group in 2024 that disrupted more than 15,000 dealerships across North America, highlighting the strategic value of dealership management platforms as access points to sensitive operational and customer data.

4.5 Off-the-Shelf Tools

The server also hosted Chisel and PuTTY plink for tunneling, Rubeus and a Mimikatz variant for credential attacks, Invoke-TheHash for pass-the-hash, ten iterative builds of Impacket's ntlmrelayx, ZeroTier VPN and OpenSSH for persistent access, and ScreenConnect for remote administration. These are standard ransomware operator tools — notable mainly because Streamer makes most of them redundant, suggesting they serve as fallbacks or predate Streamer's deployment.

5 Infrastructure exploitation

The exposed C2 server contained two exploit scripts that provide direct insight into the methods used by Eclipse operators to compromise critical infrastructure components. Both were modified versions of publicly available proof-of-concept exploits, indicating a preference for adapting existing offensive tooling rather than developing proprietary exploit capabilities. While these scripts could serve as initial access vectors against internet-facing services, they are equally suited for post-exploitation use - targeting internal infrastructure after a foothold has already been established.

5.1 VMware ESXi Compromise

One recovered script, `vmm.py`, targets CVE-2021-21974, a heap overflow vulnerability in VMware ESXi's OpenSLP service. The script was configured to exploit specific ESXi 6.7.0 builds through TCP port 427. Upon successful exploitation, it resets the root password to `Vss2026Root`, enables both SSH and the ESXi Shell, and creates a marker file (`/tmp/pwned21974`) on the compromised host.

The functionality suggests that the script was intended to establish privileged access to ESXi hosts and prepare them for follow-on operations such as deployment of the group's Linux-based encryptor. Analysis revealed only minor modifications to the original proof-of-concept published by Johnny Yu and Lucas Leong.

5.2 Veeam Backup Infrastructure Abuse

A second exploit, `poc_patched.py`, targets CVE-2024-29849, an authentication bypass vulnerability affecting Veeam Backup Enterprise Manager. The script abuses the application's SAML authentication mechanism to obtain administrative access to the web management interface without valid credentials.

The exploit was recovered alongside two PowerShell scripts, `b_veeamdec.ps1` and `b_vtypes.ps1`, which extract and decrypt credentials stored within Veeam's SQL database. Together, these artifacts demonstrate an attack chain focused on obtaining administrative access to Veeam infrastructure and recovering stored credentials from the backup environment. Compromising backup infrastructure also serves a secondary purpose common to ransomware operations: neutralizing the victim's ability to restore data without paying the ransom. The exploit itself is a modified version of a proof-of-concept originally published by Sina Kheirkhah.

5.3 Assessment

None of the exploits found on the server were zero-days, and patches were available for both vulnerabilities at the time of discovery. This shows that Eclipse operators tend to adapt publicly available exploits for operational use rather than investing in original exploit development. The exact role of these scripts in Eclipse's attack chain - whether used for initial access against exposed services or for lateral movement and privilege escalation within already compromised environments - cannot be definitively determined from the artifacts alone. However, both target high-value infrastructure components whose compromise would significantly advance a ransomware operation regardless of when in the attack chain they are employed.

6 Lateral movement and persistence

Eclipse moves laterally through a combination of built-in encryptor capabilities, Streamer modules, and operational scripts.

6.1 Encryptor Self-Spread

The Windows encryptor (locker.exe) has three lateral spread methods compiled directly into the binary — no external tooling required:

WMI: Enumerates all domain computers via ADSI, copies itself to \\target\C\$\ProgramData\, and executes remotely via Invoke-WmiMethod Win32_Process::Create. A hardcoded fallback credential (Administrator:P@sswOrd!) is embedded in the binary.

Service creation: Creates a remote Windows service named ECLIPSE[A-Z]{4} (e.g., ECLIPSEXQMB), starts it, then deletes it — a single-use execution mechanism with built-in forensic cleanup.

GPO scheduled task: The most advanced method used. Creates a Group Policy Object named "System Update Policy" with a hardcoded GUID ({CAB54552-DEEA-4691-817E-ED4A4D1AFC72}), writes a ScheduledTasks.xml with four trigger types (calendar, logon, boot, registration) repeating every five minutes, copies the payload to the NETLOGON share, and forces gpupdate on all domain computers. The task runs as SYSTEM and is hidden.

The ESXi encryptor's VM whitelist reveals how these methods work together: vCenter, pfSense, and DC01 are explicitly protected from shutdown, preserving hypervisor management, network routing, and the domain controller needed for GPO-based spread.

6.2 Operational Scripts

Recovered scripts show additional lateral techniques used during hands-on-keyboard operations: pass-the-hash via Invoke-TheHash (ITH.ps1), Kerberoasting via direct LSA API calls (krb.ps1), multi-hop WinRM pivoting through compromised Hyper-V hosts (b_basic.ps1, b_localadmin.ps1), and ten iterative builds of Impacket's ntlmrelayx for NTLM relay attacks.

```

b_basic.ps1
File Edit View

$pw = ConvertTo-SecureString 'XXXXXXXXXX' -AsPlainText -Force
$cred = New-Object System.Management.Automation.PSCredential('XXXXXXXXXX',$pw)
$out = @()
try {
    $s = New-PSSession -ComputerName '10.10.10.115' -Credential $cred -ErrorAction Stop
    $r = Invoke-Command -Session $s -ScriptBlock {
        $o = @()
        foreach($t in '10.10.10.115','10.10.10.155'){
            $o += ('=== ' + $t + ' ===')
            try {
                $res = Invoke-Command -ComputerName $t -Credential $using:cred -Authentication Basic -ErrorAction Stop -ScriptBlock
            {
                $cs = Get-WmiObject Win32_ComputerSystem
                ('HOST: ' + $cs.Name + ' | DOMAIN: ' + $cs.Domain)
                $os = Get-WmiObject Win32_OperatingSystem
                ('OS: ' + $os.Caption + ' build ' + $os.BuildNumber)
                Get-WmiObject Win32_LogicalDisk -Filter 'DriveType=3' | ForEach-Object { ($_.DeviceID + ' ' +
                [math]::Round($_.Size/1GB,0) + 'GB total ' + [math]::Round($_.FreeSpace/1GB,0) + 'GB free ' + $_.VolumeName) }
                '--- shares ---'
                Get-WmiObject Win32_Share | Where-Object { $_.Type -eq 0 } | ForEach-Object { ($_.Name + ' = ' + $_.Path) }
                '--- backup-like services ---'
                Get-Service | Where-Object { $_.Name -match 'Veeam|Backup|Arcserve|Veritas|Acronis|Shadow|DPM' } | ForEach-Object
                { ($_.Name + ' = ' + $_.Status) }
                '--- dirs C:\ ---'
                Get-ChildItem C:\ -Directory -EA SilentlyContinue | ForEach-Object { $_.Name }
            }
            $o += $res
        } catch { $o += ('basic err: ' + $_.Exception.Message.Split([char]10)[0]) }
        }
        $o -join [char]10
    }
    $out += $r
    Remove-PSSession $s
} catch { $out += ('pivot err: ' + $_.Exception.Message) }
$out -join [char]10

```

b_basic.ps screenshot

6.3 Persistence

A base64-encoded script (zt.b64) installs ZeroTier VPN, joins a specific network (743993800f45436c), deploys OpenSSH with the service set to auto-start, and opens firewall port 22 – establishing an out-of-band access channel independent of the C2 infrastructure. ScreenConnect (sv.msi) provides a secondary GUI-based remote access path.

7 Data Exfiltration

Eclipse exfiltrates data to Wasabi S3 cloud storage using per-victim buckets, credentials, and purpose-built PowerShell scripts. Three separate campaigns were recovered:

Campaign	Bucket	Target data	Notable details
Victim A - Telecommunications, Chile	[redacted]-bucket-2	130+ employee file shares across 192.168.3.x/6.x	3 script versions showing iterative refinement, plus a watchdog that auto-restarts the exfil process up to 50 times
Victim B - Education Sector, United States	[redacted]-ops-backup	23 named exfil jobs across two servers targeting student, HR, and physical security shares	Progress tracking via .done flag files, 60 retries per job
Victim C - Private Sector, Brazil	[redacted]-backup	100+ fiscal, payroll, and HR paths from a single file server	Brazilian tax records (EFD PIS/COFINS), employee admission documents

The scripts share common operational patterns: bandwidth capping (8 MB/s), file-type exclusions (executables, media, caches), and execution as SYSTEM via schtasks. The Victim A campaign is the most developed – three revisions fixed path validation bugs and removed the dependency on AWS credential profiles in favor of environment variables, suggesting the operator tested and debugged the scripts across deployments.

Beyond Wasabi, two additional exfiltration methods were observed. A stolen Google Workspace service account key (sa.json) belonging to Victim D – a private sector organization in Ireland – was used to forge three JWTs granting access to Google Drive and the Admin Directory API. A separate script (b_om5.ps1) reads an Outlook OST mailbox file using memory-mapped I/O, extracting the first 1 MB header for offline analysis – a lightweight technique for previewing mailbox contents before deciding whether to exfiltrate the full file.

8 The Encryption

Eclipse implements a hybrid post-quantum encryption scheme across all three platform variants. The architecture is consistent: Kyber768 for key encapsulation, ChaCha20 for symmetric file encryption, and a per-file random key generated via the operating system's CSPRNG. The ransom note explicitly advertises this to victims - "We use post-quantum encryption Chacha20 + Kyber, and you won't be able to decrypt your data without our help" - a deliberate signal that brute-force or cryptanalytic recovery is not an option.

```
RECOVERY_MY_NETWORK.txt x +
File Edit View H1 v ≡ v B I S ⇄ ⌂ v A6
| ECLIPSE | Ransomware |
-----
      .... Eclipse.....
    .:.                  :..
   .:.: - .. :...  :.
  .:.  ..  ....  .:.
 .:.  :.  .....  .:.
.:  :.  - *****+:::  .:.
.:  :.  *****-:::  .:.
.:  :.  =*****++:::  .:.
.:  :.  *****+:::  .:.
.:  :.  *****+:::  .:.
.:  :.  *****+:::  .:.
.:  :.  *****+:::  .:.
.:  :.  ~~~~~~.  .:.
.:  :.  .....  .:.
.:  :.  .....  .:.
.:  :.  .....  .:.
.:  :.  .....  .:.
      Ransomware

-----[ What Happened? ]-----
Your entire network has been encrypted and your data has been stolen! And all your backups have been deleted.
We use post-quantum encryption Chacha20 + Kyber, and you won't be able to decrypt your data without our help.
To restore your network and your data, you must purchase our special decryptor program and you will be able to restore your network.

-----[ What guarantees? ]-----
We understand that your data is very important to you and you want to restore your work as quickly as possible.
Therefore, we can provide you with a trial decrypt of your files (3) to ensure that our decryption program can decrypt them.

DO NOT ATTEMPT TO RECOVER THE FILES YOURSELF YOU WILL ONLY DAMAGE THEM, AND WE WILL NOT BE ABLE TO HELP YOU RECOVER THEM!!!
DO NOT CHANGE FILE EXTENSIONS!!!
DO NOT ATTEMPT TO USE DATA RECOVERY SERVICES, THEY WILL NOT BE ABLE TO DECRYPT YOUR FILES!!!
DO NOT TRY TO REPORT TO THE FBI, POLICE OR OTHER GOVERNMENT ORGANIZATIONS, THEY WILL NOT BE ABLE TO HELP YOU!!!

If you really want to recover your data, follow our instructions:

1) Download and install Tor Browser from this website: https://www.torproject.org/
2) Open your uniq Recovery Portal: http://[redacted].onion
   - to provide evidence that we have your data.
   - to test decryption of three files.
   - to delete all the stolen data after payment.
3) Enter the access key: [VICTIM_KEY]

-----[ Blog ]-----
Eclipse Blog: http://eclipse.[redacted].onion/
*data will be automatically uploaded and published on our DLS in the event of non-payment.
*after publication, they will be available to everyone and will be stored on our Tor CDN servers!
```

Ransomware note from Eclipse

For each file, the encryptor generates a fresh random key, encrypts the file contents with ChaCha20, then encapsulates the symmetric key using an embedded Kyber768 public key. The resulting ciphertext is appended to the encrypted file. Decryption requires the

corresponding Kyber768 private key, which is held exclusively by the Eclipse operators, no keys were found on the server. No implementation flaws – hardcoded keys, weak PRNG seeding, or key reuse – were identified during analysis.

The three platform variants implement this scheme independently:

	Windows (locker.exe)	Linux (linux.__rev2)	ESXi (esxi)
Language	Rust	C++	C
Kyber768	pqcrypto-kyber (Rust crate)	liboqs (C reference)	liboqs (C reference)
Symmetric	ChaCha20 via cipher crate	ChaCha20-Poly1305 via libsodium	ChaCha20 via libsodium
RNG	BCryptGenRandom	/dev/urandom via libsodium	/dev/urandom
Key format	Raw 1184 bytes in .rdata	KYPK header + 1184 bytes	KYPK header + 1184 bytes

```

ECLIPSE | Ransomware (Windows)

Usage:
  locker.exe --key <KEY> [options]

Required:
  --key <KEY>          Launch key

Targets:
  -p, --path <DIR>    Target specific path/directory

Execution Options:
  --delay <TIMESTAMP> Hold execution until specified Unix timestamp (UTC)
  --selfdelete         Automatically delete executable after execution
  --safemode           Force system reboot into Safe Mode and encrypt

Logging & Output:
  -v, --verbose        Enable console logging
  -h, --help           Show this message

```

Eclipse Ransomware CLI

The Windows encryptor offers configurable speed modes – fast, superfast, ultrafast, hyperfast, and percentage-based (10–100%) – controlling how much of each file is encrypted. Partial encryption trades recovery prevention for speed, but also means unencrypted portions of large files may be recoverable through data carving.

Anti-recovery measures are aggressive and platform-specific. The Windows variant kills 35 services (VSS, Veeam, Acronis, Sophos, SentinelAgent, SQL, Exchange) and 20+ processes

before encryption begins. The Linux variant wipes system logs, journal files, shell histories across all user profiles, disables the standard MOTD, and replaces it with the ransom note displayed on every login. The ESXi variant shuts down all virtual machines except vCenter, pfSense, and a domain controller – maximizing encryption coverage while preserving the infrastructure needed for lateral spread. All three exclude system-critical files to keep the OS bootable, and notably, the Linux variant preserves SSH keys and certificates to maintain the attackers' remote access.

9 Conclusion

The exposed Eclipse C2 server provided a rare, inside-out view of an active ransomware operation – not a snapshot of a single intrusion, but the full working inventory of a group mid-campaign.

A self-contained offensive platform. The centerpiece of Eclipse's arsenal is **Streamer**, a custom-built Rust framework consolidating over 250 offensive modules – credential theft, Active Directory reconnaissance, Kerberos attacks, lateral movement, process injection, and pivoting – into a single binary. Its embedded LuaJIT scripting engine allows operators to extend capabilities at runtime. The standalone copies of Mimikatz, Rubeus, and Chisel found alongside it appear to be legacy fallbacks largely made redundant by Streamer's native functionality.

Evasion through iteration. Eclipse's rotation across three distinct Trend Micro products as payload carriers – progressing from 23/71 detections on the earliest variant to zero submissions for the latest – reveals a group actively monitoring and adapting its delivery mechanisms in response to detection.

Infrastructure as a target. Exploit scripts for VMware ESXi and Veeam Backup Enterprise Manager, paired with credential-extraction tools for Veeam's SQL database, show Eclipse systematically targeting the infrastructure organizations depend on for resilience – expanding access while eliminating recovery options before encryption begins.

Structured exfiltration via Wasabi. Data theft is built around Wasabi S3 cloud storage – a provider favored by a small number of ransomware operators, including LockBit affiliates, for its low cost, lack of egress fees, and S3-compatible API. Eclipse's use is notably more structured than previously documented cases: dedicated buckets per victim, bandwidth-capped transfers to avoid triggering network alerts, automated retry and watchdog mechanisms, and iterative script refinement across engagements. This consistent, disciplined use of Wasabi as a primary exfiltration channel is a distinguishing operational characteristic of the group.

Sound encryption with no identified flaws. Kyber768 paired with ChaCha20 is implemented consistently across all three platform variants with per-file random keys. No private keys were recovered, and no implementation flaws were identified during static analysis.

What this means for defenders. This analysis establishes a baseline of Eclipse's tooling and tradecraft as of October 2026. Because the recovered toolkit represents the group's full operational capability rather than fragments from a single incident, defenders can use it not only to identify ongoing compromises but to anticipate the techniques Eclipse is likely to employ next. Organizations that validate their detection coverage against the indicators documented here – particularly around backup infrastructure access, anomalous S3 traffic, LSASS interaction from unsigned binaries, and GPO-based propagation mechanism – will be better positioned to disrupt an Eclipse intrusion before encryption begins.

10 Indicators of Compromise

10.1 Network Indicators

Indicator	Log source	Detail
HTTPS to `s3.wasabisys.com`, `s3.us-west-1.wasabisys.com`, `s3.us-east-2.wasabisys.com`	Proxy logs, firewall logs, or DNS logs if proxy visibility is limited	Internal servers connecting to Wasabi S3 – unusual in most enterprise environments
Connections to `46.165.212.179`	Firewall logs, netflow	C2 staging server
ZeroTier traffic (UDP 9993 to ZeroTier root servers)	Firewall logs	Unexpected P2P VPN traffic from endpoints
SSH (port 22) outbound from Windows hosts	Firewall logs	Windows servers initiating SSH is unusual in most environments
Tor traffic or connections to known Tor entry nodes	Proxy / firewall logs, threat intel feeds	Victim portal and DLS communication

10.2 Windows Event Log Indicators

Indicator	Event ID / Log	Detail
Service creation matching `ECLIPSE[A- Z]{4}`	Event ID 7045 (System)	Service name field matches pattern; `type=own, start=auto`
GPO created with GUID `{CAB54552- DEEA-4691-817E- ED4A4D1AFC72}`	Event ID 5136 (Security – Directory Service Changes)	Requires AD audit policy for DS changes enabled
GPO named "System Update Policy"	Event ID 5136 (Security)	Same – monitor `displayName` attribute
Scheduled task "Windows System Update Service"	Event ID 4698 (Security)	Task creation audit; runs as SYSTEM, hidden
Bulk service stop events for VSS, Veeam, SQL, Exchange	Event ID 7036 (System)	Multiple critical services entering "stopped" state within seconds
`sshd` service set to auto-start	Event ID 7045 (System)	OpenSSH server installed as persistence mechanism

<code>`aws.exe` process with `--endpoint-url *wasabisys*` in command line</code>	Event ID 4688 (Security) with command-line logging enabled, or Sysmon Event ID 1	Process creation with full command line
PowerShell executing <code>`s3 sync`</code> or <code>`s3 cp`</code> with Wasabi endpoints	PowerShell ScriptBlock Logging (Event ID 4104)	Requires ScriptBlock logging enabled
WMI remote process creation via <code>`Invoke- WmiMethod`</code>	Event ID 4688 on target host, WMI-Activity/Operational log	Lateral spread – look for <code>`Win32_Process::Create`</code>
<code>`sc.exe create`</code> with service name starting <code>`ECLIPSE`</code>	Sysmon Event ID 1 / Event ID 4688	Lateral spread via service creation

10.3 EDR / Sysmon Indicators

Indicator	Sysmon Event ID	Detail
File creation: <code>`RECOVERY_MY_NETWORK.txt`</code>	Event ID 11 (FileCreate)	Ransom note – late-stage
File creation: <code>`eclipse`</code> extension in bulk	Event ID 11 (FileCreate)	Encryption in progress
File creation: <code>`k.log`</code> or <code>`k2.log`</code> in <code>`C:\ProgramData\kl32\`</code>	Event ID 11 (FileCreate)	Keylogger output
File creation: <code>`pn_marker.txt`</code> in <code>`C:\ProgramData\b8f3a1c2\`</code>	Event ID 11 (FileCreate)	Staging artifact
Process creation: <code>`cmd.exe /c *pn.cmd`</code>	Event ID 1 (ProcessCreate)	Execution of batch script
Process loading DLL with Trend Micro version info from <code>`C:\ProgramData\`</code>	Event ID 7 (ImageLoaded)	Weaponized TM binary outside any TM installation
LSASS access from unsigned process in <code>`C:\ProgramData\`</code> subdirectory	Event ID 10 (ProcessAccess) with <code>`TargetImage` = `lsass.exe`</code>	Streamer credential theft
<code>`SetWindowsHookExA`</code> with <code>`WH_KEYBOARD_LL`</code> from unsigned binary in <code>`C:\ProgramData\`</code>	EDR behavioral detection	Keylogger hooking – combine with

		path for low false-positive
Binary with self-signed cert (thumbprint `2A048197B529B77C66F42D0DBDB2F12D5481773F`)	Event ID 7 (ImageLoaded) with signature metadata	Code-signed trxHandler variant

10.4 Active Directory Indicators

Indicator	Log source	Detail
ADSI query `objectCategory=computer` enumerating all domain computers	Domain controller Security log, Event ID 4662	Encryptor pre-spread reconnaissance
Payload copied to `NETLOGON` share	Domain controller Security log, Event ID 5145 (file share access)	GPO lateral spread staging
`gpupdate /force` executed across multiple hosts simultaneously	Sysmon Event ID 1 on target hosts	Forced GPO application after malicious policy creation

10.5 Linux / ESXi Indicators

Indicator	Log source	Detail
Marker file `/tmp/pwned21974`	File integrity monitoring	ESXi exploitation marker
Password `Vss2026Root` on ESXi host	Post-compromise audit	Set by CVE-2021-21974 exploit
Mass `vim-cmd vmshvc/power.off` commands	ESXi shell history (if not wiped), hostd.log	ESXi encryptor shutting down VMs
`chmod -x /etc/update-motd.d/*` followed by `chmod +x /etc/profile.d/eclipse.sh`	File integrity monitoring, auditd	Linux ransom note persistence
Bulk deletion of `/var/log/journal/*`, `/var/log/*.gz`	auditd (if configured before wipe)	Anti-forensic log destruction
`systemctl disable` targeting multiple services in rapid succession	journald (if captured before wipe)	Linux encryptor pre-encryption service kill

10.6 File Hashes

SHA-256	File
`446D94FEBD47A77E822A1A2A45E7E4EDCOE873C67FF2DDBE65CFC3034A CF28F5`	locker.exe (Windows encryptor)
`889367F8A72EB40E1874320678F829BCBD04D4684C2D353D275D810629A E1C29`	linux.__rev2 (Linux encryptor)
`7F9CB5A5B2B004B39D3B72FE58B399FC4C3C735C037E76C30FC5C88B4 6285E87`	esxi (ESXi encryptor)
`5F608E1F074E33E71163196D63ADB7B14A181958A37E39446EA8E843AA9BD A4E`	winsvc.exe (weaponized ATSE64)
`CF2B3E3001090A3486B698DE2D76A5CE67B719587E6B1BE4B883D14D736F D482`	rust.exe (weaponized trxHandler)
`DE4648C5921F05E61ABE4910AD20752FD9CD7BD1828C541AB0A44974802 57091`	rust_signed.e xe (code- signed trxHandler)
`8E388035E96E8976DA09BB3A4C2D429B40686D94938A5D07AE51D67882 D3B1AA`	122.exe (weaponized TmopsmHttp)
`527C0E5F7D85909263B6E6D37EFC06ADD1362E8458284C311C7B252DF08 47CFD`	b3f7a1c9.exe (weaponized trxHandler, second build)
`EFFACD91B29715B1B1228DB18A0ACA99B57D810D0228CDCF2A4F39D36F4 CDE5B`	kl.exe (CDK- targeted keylogger)
`8047A062A9B65889F1F78C63F6A1E3AF9FA6DB44B2D716F3DC4C8846097 E4E9A`	kl2.exe (general keylogger)
`5EDD67678D0956B0C4BAF7A3E3A0B2424D04DEBD5734BFE5CC1B19C3D 4BDFE47`	evil.dll

`E885B1978D05E454E774B727EE7B1D076E3B2B46B8B3F30B99DCA7431FBD0068`	evil2.dll
`FBD309CCF8EB111EC5074BDCC9F83E2E896460C516B90DF99853075A8BB047E6`	Streamer beacon (decrypted from winsvc.exe)

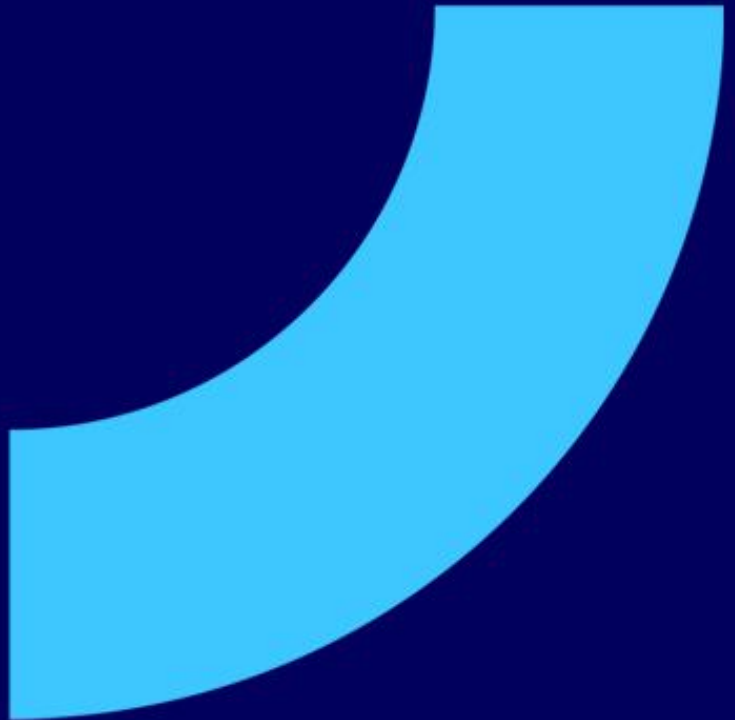
About Atos Group

Atos Group is a global leader in digital transformation with c. 56,000 employees and annual revenue of c. €7.2 billion (at the go-forward perimeter), operating in 54 countries under two brands - Atos for services and Eviden for products and systems. European number one in cybersecurity and a leader in cloud, Atos Group is committed to a secure and decarbonized future and provides tailored AI-powered, end-to-end solutions for all industries. Atos Group is listed on Euronext Paris.

Find out more about us

atos.net

atos.net/career



Atos Group is a registered trademark of Atos Group. © Atos Group. Confidential Information owned by Atos Group, to be used by the recipient only. This document, or any part of it, may not be reproduced, copied, circulated and/or distributed nor quoted without prior written approval of Atos Group.

Atos