Everest

Technical Analysis — Cti Report

REVERSE-ENGINEERED REPORT

RansomLook · ransomlook.io
File last modified: 2026-05-15

Sample SHA-256: `1df92bf4c967297d8a39fc3f619a56702ee96d5cf9196b8e1d5b3654746c6514`

> **Scope statement:** this analysis covers a single protected sample (SHA-256 in §1). Findings describe what was observed in this binary; they do not imply that all Everest deployments share identical implementation details. Family-level claims are restricted to indicators flagged as Family in §12.
>
> **Reading convention:** the markers `**Observed:**`, `**Inferred:**`, and `**Hypothesis (not demonstrated):**` are used throughout this brief to separate facts read directly from the binary from analytical extrapolation.

1. Executive Summary

Family	Self-identified as EVEREST (via `.everest` extension, `EVERESTRANSOMWARE.txt` note filename, `"Greetings from the Everest team"` salutation, `everestaf@onionmail.org` contact, blog onion URL)
Sample variant	Per-deployment build (note in `<Module>.smethod_5` carries a hard-coded victim salutation; RSA-1024 public key in `<Module>.smethod_19` is sample-unique)
Platform	Windows .NET 4.0 (PE32 i386 managed, ConfuserEx-protected)
Builder fingerprint	ConfuserEx 1.x with `rename` + `constants` + `anti-tamper` + `compressor` presets, watermark stripped (DIE: "Modified managed EP + Int confusion + Short names + Bad .cctor format")
Encryption	AES-128-CBC PKCS#7 (small files) / AES-128-CBC NoPadding intermittent (>10 MB), key+IV derived via PBKDF2-HMACSHA1 from a 32-byte ASCII printable seed wrapped with **RSA-1024 PKCS#1 v1.5**
Concurrency	3 background worker threads (anti-RE/RE-tool kill loop 4 s, service+process kill loop 15 s, memory-pig kill loop 2.5 s) plus the main thread
Discovery	5 LAN-enumeration sources (`net view` , `NetDfsEnum` , `WNetEnumResource` , WMI `Win32_Share` , WMI `Win32_NetworkConnection` + `Win32_MappedLogicalDisk`); ARP-cache parsing
Pre-encryption operations	Wake-On-LAN UDP magic packet broadcast (ports 7 and 9), `mountvol.exe` of unlettered volumes, Restart Manager force-release of file handles, process self-DACL deny-Everyone
Cryptographic primitives	RSA-1024 PKCS#1 v1.5 (declared `RSACryptoServiceProvider(4096)` but `FromXmlString` overrides with a 128-byte modulus); AES-128-CBC (declared `KeySize=256` but `Key=byte[16]` overrides); PBKDF2-HMACSHA1 with static 8-byte salt and 1000 iterations; `System.Random` for the 32-byte ASCII seed; same Key+IV reused for every file in one run
Exfiltration	The note text claims ~1 TB exfiltration. **No exfiltration code is present in this binary** (no socket/HTTP-POST/WinHTTP/Ws2_32 imports). The claim is text-only
Recovery (operator-side)	RSA-wrapped per-sample seed is persisted in `HKCU\Software\AppName\PublicKey` and inside the dropped note
Recovery (victim-side)	The seed is generated by `System.Random` (boot-tick-seeded by default) and the PBKDF2 salt is static — both are observed facts. **Whether either creates an exploitable recovery path against an unknown private RSA key is not demonstrated in this analysis**
Sample SHA-256	`1df92bf4c967297d8a39fc3f619a56702ee96d5cf9196b8e1d5b3654746c6514`
Onion blog	`ransomocmou6mnbquqz44ewosbkjk3o5qjsl3orawojexfook2j7esad.onion`
Email	`everestaf@onionmail.org`

2. Notable Observations

2.1 Bespoke binary per engagement

Observed: the decoded note in `<Module>.smethod_5` begins with "Dear <name>," where the name is a hard-coded literal embedded in the encrypted UserStrings heap. The RSA public key in `<Module>.smethod_19` (modulus listed in §3) is also sample-unique. Both items are part of the same build.

Inferred — operational implication: blocking by SHA-256 alone is insufficient for prevention. Threat hunting must rely on the structural and behavioural artefacts that are stable across builds (mutex GUID format, extension, note filename, command sequences, ConfuserEx fingerprint). The RSA modulus and the salutation string are pivots for the **specific sample**, not for the family at large.

2.2 Misleading crypto declarations

Observed: the code declares cryptographic primitives at twice the strength they actually use.

Declared in source	Actual at runtime	Cause
`new RSACryptoServiceProvider(4096)`	RSA-1024	`FromXmlString(<XML>)` overrides the constructor's keysize with the embedded 128-byte modulus
`aes.KeySize = 256` (set first)	AES-128	`aes.Key = byte[16]` setter (called second) silently downgrades KeySize to 128

The runtime values are 1024-bit RSA and 128-bit AES regardless of the constructor arguments. This is a property of how `RSACryptoServiceProvider.FromXmlString` and `AesCryptoServiceProvider.Key` setters behave when called after the size declaration.

2.2.1 Cryptographic weakness inventory — three-tier separation

This table separates observed weakness, theoretical security impact, and practical exploit demonstrated in this analysis. The third column prevents misreading "weak primitive" as "decryptable".

Weakness	Observed	Practical exploit demonstrated?
Declared RSA-4096 actually RSA-1024	Yes — `FromXmlString` overrides keysize, modulus is 128 bytes	**No** — 1024-bit factoring remains computationally infeasible without nation-state hardware
Declared AES-256 actually AES-128	Yes — `Key=byte[16]` silently downgrades KeySize	**No** — AES-128 is still cryptographically intact
32-byte AES seed from `System.Random`	Yes — `Random.Next(33,127)`, default seed = `Environment.TickCount` at first instantiation	**No** — would require: (a) tick-precision boot timestamp, (b) all 32 outputs reconstructible from one seed, (c) plus the RSA private key. Hypothesis only
Static PBKDF2 salt `01 02 03 04 05 06 07 08`	Yes — hard-coded in `Program.Db`	**No** — static salt enables rainbow tables against passwords, but PBKDF2 input here is the 32-byte ASCII seed (already entropy-bearing); salt does not break the primitive
AES Key+IV reused across all files in one run	Yes — both static fields, set once per process lifetime	**No** — IV reuse with CBC weakens semantic security but does not yield plaintext recovery without same-prefix correlation across files; no exploit pipeline shown
RSA-1024 PKCS#1 v1.5 wrapping	Yes — block size 128 B, no OAEP	**No** — Bleichenbacher-style oracles would require operator-side decryption interaction (not in scope of victim-side analysis)

2.3 PRNG choice for the seed

```
char c = (char) random.Next(33, 127);   // System.Random
```

Observed: the 32-byte AES seed is generated by `System.Random`, not by `RNGCryptoServiceProvider`. `System.Random` is documented by Microsoft as not suitable for cryptographic use; the default seed is `Environment.TickCount` at first instantiation. The PBKDF2 salt used downstream is the static 8-byte sequence `01 02 03 04 05 06 07 08` (in `Program.Db`). The PBKDF2 iteration count is 1000.

Hypothesis (not demonstrated): if the boot tick can be reconstructed to ms precision from a captured Event Log on the victim host, the 32-byte seed sequence is in principle reproducible — but this would only recover the AES Key+IV from a chain of Random outputs, and the RSA-wrapped seed (the operator-side recovery token) remains untouched. No exploit chain end-to-end is provided.

2.4 Wake-On-LAN broadcast

Observed: `WakeOnLan.SendMagicPacket(MAC, IP)` constructs a magic packet (`0xFF×6 + MAC×16` = 102 bytes) and broadcasts it on UDP ports 7 and 9 to the calculated subnet broadcast address (IP | ~mask, mask hard-coded `255.255.255.0`). Targets are sourced from the local ARP cache via `arp -a` parsing in `ArpParser.ParseArpTable`.

Order in the spread sequence (observed): 1. `arp -a` parse → list of `(IP, MAC)` pairs 2. For each pair: WoL packet to `(IP | ~mask)` on UDP/7 and UDP/9 (synchronous loop, no rate limiting) 3. Then SMB enumeration begins (`net view` → `NetDfsEnum` → `WNetEnumResource` → WMI `Win32_Share` / `Win32_NetworkConnection` / `Win32_MappedLogicalDisk`) 4. Then `net use \\<unc>` per discovered share

Fan-out (observed): one call per ARP entry × 2 ports = 2 packets per host. No broadcast aggregation. Frequency is bounded by the loop's I/O throughput on `Socket.SendTo`, typically ms-scale per packet.

Timing relative to encryption (observed): invocation occurs in `LanPropagation.DiscoverAndMountShares`, which runs during EP setup before the encryption pass — approximately T-15 s before the first file destruction (see §4 Execution Timeline).

UDP/7 and UDP/9 outbound broadcast traffic with a 102-byte payload is a stable observable for this code path and a high-precision NDR/Zeek pivot.

2.5 Process self-protection via DACL

Observed: `KZsyzkgZlDQw.UB()` modifies the running process's security descriptor to insert an `AccessDenied` ACE for the well-known World SID (S-1-1-0):

```
RawSecurityDescriptor sd = GetKernelObjectSecurity(GetCurrentProcess(), DACL);
sd.DiscretionaryAcl.InsertAce(0,
  new CommonAce(AceFlags.None, AceQualifier.AccessDenied,
                accessMask: 0x1F0FFF, sid: WorldSid, ...));
SetKernelObjectSecurity(GetCurrentProcess(), DACL, sd);
```

Effect: standard user-mode termination paths (including common SYSTEM-context tooling such as `taskkill /F`) fail with Access Denied against this process. Termination requires re-acquiring `WRITE_DAC` rights on the handle (typically via `SetSecurityInfo` from the owner SID, or via a kernel-mode driver / `SeDebugPrivilege` flow that bypasses user-mode DACL evaluation). Modern EDR with kernel callbacks (`PsSetCreateProcessNotifyRoutineEx`, `ObRegisterCallbacks`) and `SeTakeOwnershipPrivilege` can defeat this; legacy AV and many response-tool kill chains do not attempt the DACL repair step. **Strong anti-tamper, not absolute.**

2.6 Lateral movement broadens the attack surface BEFORE encryption

Observed sequential operations during EP setup (before any file is encrypted):

1. `Set-MpPreference -EnableControlledFolderAccess Disabled` (Windows 10/11)
2. `Enable-WindowsOptionalFeature -Online -FeatureName SMB1Protocol`
3. `netsh advfirewall firewall set rule group="File and Printer Sharing" new enable=Yes`
4. `netsh advfirewall firewall set rule group="Network Discovery" new enable=Yes`
5. `sc config Dnscache start= auto` (and 4 other discovery services), then `ServiceController.Start`
6. `reg ADD ...\LongPathsEnabled = 1`
7. `reg ADD ...\LocalAccountTokenFilterPolicy = 1`
8. `reg ADD ...\EnableLinkedConnections = 1`
9. `icacls "C:*" /grant Everyone:F /T /C /Q` (and D:, Z:)
10. Mount unlettered volumes via `mountvol.exe`

Inferred: the combination of **SMB1 re-enabled + LocalAccountTokenFilterPolicy + EnableLinkedConnections** opens the host significantly. Defenders monitoring registry/firewall change events have a clear pre-encryption signal window (typically 30–120 seconds, derived from the observed thread spin-up latency in §4).

2.7 Build-template residual artifact

Observed: the wallpaper file path string is `\[WALLNAME].bmp` — the literal `[WALLNAME]` placeholder is present **as-is** in the binary's strings; it is not substituted to a real filename at build or runtime. This is an unsubstituted template token.

Inferred: as a hunting indicator, the literal `[WALLNAME]` is observable in any sample where the same template artifact is preserved; this is a build-line artefact rather than a per-sample one.

2.8 Engineering anomalies

- The `gj8BnGv30aeJ.Contains` check at the EP uses a string returned by `XXGcXDwCBmxY.tb()` (WMI `Win32_OperatingSystem.Caption + OSArchitecture`). The presence of `"10"` is checked to gate the Defender CFA bypass, but `"10"` also matches **Windows Server 2010**, **Windows 10**, **Windows 11**, **build numbers containing 10**, etc. — an over-broad heuristic.
- The ARP-derived RFC1918 filter (`huHoNzPzkzhd.Oc.StartsWith(huHoNzPzkzhd.Oc)`) is **buggy** — the `StartsWith` argument is the same field, making the check tautological. Effect: **all ARP-discovered hosts are processed**, including non-RFC1918 (public IPs in the cache), broadening the WoL blast radius.
- The string `[FULL]` is the sentinel used in `Program.ExtensionWhitelist[0]` to signal "no extension filter (encrypt all)"; `[auto]` is the sentinel in `Program.DriveWhitelist[0]` for "use `DriveInfo.GetDrives()` (all drives)". These are literal sentinels in the binary; their value is checked by `Encryptor.EncryptAll`.

---

3. Code-level fingerprints

Stable artefacts in the binary, observable across builds of this family:

Marker	Where in binary
Note filename literal `EVERESTRANSOMWARE.txt`	`<Module>` getter, decoded via reflection
File extension `.everest`	Same
Decoded note opens with `"Greetings from the Everest team"`	`<Module>.smethod_5`
Contact email `everestaf@onionmail.org`	Note text
Onion blog URL `ransomocmou6mnbquqz44ewosbkjk3o5qjsl3orawojexfook2j7esad.onion`	Note text
Mutex `Global\7efc73f7-fda1-42d1-a4c5-8f1670bd08a5`	EP, before single-instance check
Wallpaper template literal `[WALLNAME].bmp` (unsubstituted)	`<Module>` getter
ConfuserEx 1.x watermark stripped (DIE: Modified managed EP + Int confusion + Short names + Bad .cctor format)	PE/metadata-level
GZip+Base64+lookup string decryptor `<Module>.m(string)`	Decryptor body
Static PBKDF2 salt `01 02 03 04 05 06 07 08`	`Program.Db` field initializer
`Random.Next(33,127)` for 32-byte seed (not RNGCryptoServiceProvider)	`Helpers.GenSeed32Random`
Wake-On-LAN UDP magic packet broadcast on ports 7 and 9	`WakeOnLan.SendMagicPacket`
Per-host file in `C:\ProgramData\<MD5(ProcessorId+VolumeSerial)>` listing scanned UNCs	`LanPropagation.DiscoverAndMountShares`
Note text typos: `"trully"`, `"singifically"`, `"backups , etc. etc."`	Note body

Sample-specific (non-stable across builds):

Marker	Source
RSA-1024 public key (XML form, base64)	`<Module>.smethod_19`
Hard-coded victim salutation in note	`<Module>.smethod_5`

The salutation token in this sample is redacted in this report. The binary contains the literal value as part of the encrypted strings heap.

3.1 CIS-exclusion list (factual content)

The geo-fence at the EP compares `CultureInfo.InstalledUICulture.Name` and `CultureInfo.CurrentCulture.LCID` against the following lists (decoded from `<Module>.smethod_*`):

Type	Values
Cultures (string)	`hy-AM, az-Cyrl-AZ, Cy-az-AZ, Lt-az-AZ, be-BY, kk-KZ, ky-KZ, ky-KZ, tt-RU, ba-RU, sah-RU, ru-RU, tg-Cyrl-TJ, uz-Cyrl-UZ, uk-UA, ka-GE`
LCIDs (decimal)	`1049, 1067, 2092, 1068, 1059, 1079, 1087, 1064, 1090, 2115, 1091, 1058`

On any match, `Program.Exit()` is called (`Process.Kill` + `Environment.Exit(0)`).

---

4. Execution Timeline (T0 → encryption)

Relative-order timeline (sequential dependencies in the observed code path; absolute timings are approximate, drawn from thread spin-up latency in a representative run).

```
T+0.0   Mutex Global\7efc73f7-fda1-42d1-a4c5-8f1670bd08a5 created (single-instance gate)
T+0.1   CIS culture/LCID check                — Process.Kill + Environment.Exit(0) on match
T+0.5   Self-DACL deny-Everyone ACE inserted  — KZsyzkgZlDQw.UB() (anti-tamper)
T+1.0   Set-MpPreference -EnableControlledFolderAccess Disabled (Win10/11 only)
T+2.0   Enable-WindowsOptionalFeature -Online -FeatureName SMB1Protocol (DISM)
T+3.0   netsh advfirewall — File and Printer Sharing + Network Discovery enable
T+4.0   reg ADD LongPathsEnabled / LocalAccountTokenFilterPolicy / EnableLinkedConnections
T+5.0   sc config Dnscache + Fdrespub + FDPHost + SSDPSRV + UPnPHost (start=auto)
        ServiceController.Start on each
T+6.0   icacls "C:*" /grant Everyone:F /T /C /Q   (then D:*, Z:*)
T+7.0   mountvol.exe enumeration + mount of \\?\Volume{...} (unlettered volumes)
T+8.0   arp -a parsed by ArpParser.ParseArpTable    [Discovery starts]
T+9.0   Wake-On-LAN UDP/7 + UDP/9 broadcast burst (per ARP entry × 2 ports)
T+10.0  Share enumeration — 5 sources in sequence:
        net view → NetDfsEnum → WNetEnumResource → WMI Win32_Share
        → WMI Win32_NetworkConnection + Win32_MappedLogicalDisk
T+12.0  net use \\<unc> per discovered share (one-shot mount; failures silent)
        Per-host UNC list persisted to C:\ProgramData\<MD5(ProcessorId+VolumeSerial)>
T+15.0  Restart Manager session opened — RmRegisterResources + RmGetList
        (force-release of locked file handles)
T+16.0  Worker thread #1 spawn — RE-tool kill loop (4 s interval)
T+16.0  Worker thread #2 spawn — service+process kill loop (15 s interval)
T+16.0  Worker thread #3 spawn — memory-pig kill loop (>250 MB threshold, 2.5 s)
T+18.0  vssadmin Delete Shadows /all /quiet
        OR (PowerShell path) Get-CimInstance Win32_ShadowCopy | Remove-CimInstance
T+18.5  Srclient!SRRemoveRestorePoint loop over all restore points
T+19.0  Anti-Raccine cleanup — IFEO entries, HKLM\SOFTWARE\Raccine,
        HKCU\…\Run\Raccine Tray, scheduled task "Raccine Rules Updater",
        taskkill Raccine.exe / RaccineSettings.exe
T+20.0  cmd.exe /c del /s /f /q against backup file patterns
        (*.VHD, *.bak, *.bkf, etc.) on drives C-H
T+22.0  Wallpaper write to %TEMP%\[WALLNAME].bmp
        Registry SetValue HKCU\Control Panel\Desktop\WallPaper
T+23.0  Encryption pass begins — Encryptor.EncryptAll
         ├ Per file: PBKDF2(seed, salt=Db, iter=1000) → AES-128 Key+IV
         ├ Files >10 MB: intermittent NoPadding chunked encryption
         ├ Files <10 MB: full PKCS#7 CBC pass
         └ File renamed to <orig>.everest
T+N     Note drop — EVERESTRANSOMWARE.txt in C:\ProgramData and Desktop
        Registry write — HKCU\Software\AppName\PublicKey = <Base64 wrapped seed>
T+N+1   Self-delete — cmd.exe /C ping 127.0.0.7 -n 3 > Nul
                    & fsutil file setZeroData offset=0 length=<size> <path>
                    & Del /f /q <path>
```

Defender takeaway: the **first three commands (T+1 → T+3)** provide a 30+ second action window before any file is destroyed. The DACL ACE at T+0.5 means user-mode kill commands issued after that point will fail unless the responder re-acquires `WRITE_DAC`.

5. TTP Highlights for Defender Workflow

5.1 Pre-encryption indicators (high precision, low FP)

Signal	Source	Time before file destruction
`Global\7efc73f7-fda1-42d1-a4c5-8f1670bd08a5` mutex creation	EDR (Sysmon Event 17 / kernel ETW)	~immediate
`Set-MpPreference -EnableControlledFolderAccess Disabled`	PowerShell ScriptBlockLogging (Event 4104) or Microsoft-Windows-PowerShell/Operational	T-30 to T-60 s
`Enable-WindowsOptionalFeature SMB1Protocol`	Same	T-30 to T-60 s
`netsh advfirewall ... new enable=Yes` (group=Network Discovery)	Sysmon Event 1 (process creation)	T-30 to T-60 s
`mountvol.exe` invoked with `\\?\Volume{...}` in command line	Sysmon Event 1	T-25 s
`arp -a` invoked from a non-shell parent	Sysmon Event 1	T-20 s
Outbound UDP/7 or UDP/9 broadcast (102-byte payload)	NetFlow / firewall logs	T-15 s
`vssadmin.exe Delete Shadows /all /quiet` OR `Remove-CimInstance` on Win32_ShadowCopy	Sysmon Event 1 / 4104	T-10 s
`cmd.exe /c rd /s /q %SYSTEMDRIVE%\$Recycle.bin`	Sysmon Event 1	T-5 s
`icacls "C:*" /grant Everyone:F /T /C /Q`	Sysmon Event 1	T-3 s

The **first three** events (CFA disable, SMB1 enable, netsh) provide a **30+ second action window** for defenders.

5.2 During-encryption indicators

Signal	Source
`cmd.exe /c del /s /f /q <pattern>` for `*.VHD`, `*.bak`, etc. on multiple drives	Sysmon Event 1
Service stop wave (~100 services in ~30 s)	Service Control Manager Event 7036 (state changes)
Process termination wave matching DBs/AV/backup names	Sysmon Event 5 (process termination)
File rename/create with `.everest` extension	EDR file telemetry / FIM
Wallpaper update event	Sysmon Event 12 (registry SetValue on `HKCU\Control Panel\Desktop\WallPaper`)

5.3 Post-encryption indicators

Signal	Source
`EVERESTRANSOMWARE.txt` file present in `C:\ProgramData` and on Desktop	FIM / EDR scan
`\[WALLNAME].bmp` file in `%TEMP%`	FIM
`HKCU\Software\AppName\PublicKey` exists with a Base64 DWORD blob (~684 chars)	Registry telemetry
`C:\ProgramData\<32-hex-chars>` file (no extension) containing UNC paths	FIM
`cmd.exe /C ping 127.0.0.7 -n 3 > Nul & fsutil file setZeroData ... & Del /f /q ...`	Sysmon Event 1

---

6. MITRE ATT&CK Matrix

Consolidated mapping of techniques observed in this binary. Tactics not represented (Initial Access, Collection, Command and Control, Exfiltration as code) are listed for completeness.

Tactic	Technique	ATT&CK ID	Notes
Initial Access	(out of scope)	—	Loader/initial vector not present in this binary
Execution	Command and Scripting Interpreter: PowerShell	T1059.001	`Set-MpPreference`, `Enable-WindowsOptionalFeature`, `Get-CimInstance`
Execution	Command and Scripting Interpreter: Windows Cmd	T1059.003	Multiple `cmd.exe /c` chains
Execution	Native API	T1106	`RmRegisterResources`, `SetKernelObjectSecurity`, `WMI`
Persistence	Modify Registry	T1112	`HKCU\Software\AppName\PublicKey` write (recovery token, not autorun)
Defense Evasion	Impair Defenses: Disable or Modify Tools	T1562.001	CFA disable; service kill loop targets AV/EDR processes
Defense Evasion	Modify Registry	T1112	`LongPathsEnabled`, `LocalAccountTokenFilterPolicy`, `EnableLinkedConnections`
Defense Evasion	Hide Artifacts	T1564	DACL self-protection — process inaccessible to standard kill paths
Defense Evasion	Modify Authentication Process	T1556	`LocalAccountTokenFilterPolicy = 1`
Defense Evasion	Indicator Removal: File Deletion	T1070.004	Self-delete chain: `fsutil setZeroData` + `Del /f /q`
Defense Evasion	Obfuscated Files or Information	T1027	ConfuserEx — string encryption, anti-tamper, constants module
Discovery	System Network Configuration Discovery	T1016	`arp -a` parsing
Discovery	System Network Connections Discovery	T1049	`net view`, `WNetEnumResource`
Discovery	Remote System Discovery	T1018	`NetDfsEnum`, ARP-derived target list
Discovery	Network Share Discovery	T1135	WMI `Win32_Share`, `Win32_NetworkConnection`, `Win32_MappedLogicalDisk`
Discovery	System Information Discovery	T1082	WMI `Win32_OperatingSystem` (caption + arch) for OS gating
Discovery	System Location Discovery	T1614.001	`CultureInfo.InstalledUICulture`, LCID check (CIS exclusion)
Lateral Movement	Remote Services: SMB/ Windows Admin Shares	T1021.002	`net use \\<unc>` per discovered share
Lateral Movement	(custom) Wake-On-LAN assist	—	Non-standard — no canonical ATT&CK technique. Tracked as pre-spread enabler
Collection	(out of scope)	—	No collection routine in this binary
Command and Control	(out of scope)	—	No C2 channel observed
Exfiltration	(claimed in note text only)	—	Note states ~1 TB exfil; no exfil code in binary

Tactic	Technique	ATT&CK ID	Notes
Impact	Data Encrypted for Impact	T1486	AES-128-CBC pass over file content
Impact	Inhibit System Recovery	T1490	`vssadmin Delete Shadows`, `SRRemoveRestorePoint`, backup file deletion
Impact	Service Stop	T1489	Worker thread #2 — service+process kill loop
Impact	Defacement: Internal	T1491.001	Wallpaper change to `[WALLNAME].bmp`
Impact	File and Directory Permissions Modification	T1222.001	`icacls "C:*" /grant Everyone:F /T /C /Q` (and D:, Z:)

7. Hunting Queries

7.1 Splunk SPL — Pre-encryption window detection

```
index=sysmon EventCode=1
| where (
    (CommandLine="*Set-MpPreference*EnableControlledFolderAccess Disabled*")
    OR (CommandLine="*Enable-WindowsOptionalFeature*SMB1Protocol*")
    OR (CommandLine="*advfirewall*Network Discovery*new enable=Yes*")
    OR (CommandLine="*advfirewall*File and Printer Sharing*new enable=Yes*")
    OR (CommandLine="*mountvol*\\\\?\\Volume{*")
)
| stats count, values(CommandLine) by host, ParentImage
| where count >= 3
```

Interpretation: any host running ≥3 of these enabling commands within a sliding window is in the pre-encryption setup phase of an Everest-style ransomware (consistent with observed Everest tradecraft; not exclusive to this family).

7.2 Splunk SPL — Post-incident DFIR pivot

```
index=sysmon EventCode=11 (TargetFilename="*\\EVERESTRANSOMWARE.txt" OR
TargetFilename="*.everest")
| eval first_drop = if(TargetFilename LIKE "%\\EVERESTRANSOMWARE.txt", _time, null)
| stats min(first_drop) as note_drop_time, count(eval(TargetFilename LIKE "%.everest")) as
encrypted_count by host
```

7.3 KQL — Microsoft Defender Advanced Hunting

```
DeviceProcessEvents
| where Timestamp > ago(7d)
| where (
    (ProcessCommandLine has "Set-MpPreference" and ProcessCommandLine has "EnableControlledFolder
Access" and ProcessCommandLine has "Disabled")
    or (ProcessCommandLine has "SMB1Protocol")
    or (ProcessCommandLine has "advfirewall" and ProcessCommandLine has "Network Discovery")
    or (ProcessCommandLine has "mountvol" and ProcessCommandLine has "Volume{")
    or (ProcessCommandLine has "icacls" and ProcessCommandLine has "Everyone:F" and ProcessComman
dLine has "/T /C /Q")
)
| summarize Pre_Indicators = make_set(ProcessCommandLine), Indicator_Count = dcount(ProcessComman
dLine)
            by DeviceName, bin(Timestamp, 5m)
| where Indicator_Count >= 3
| order by Timestamp desc
```

7.4 KQL — DACL self-protection signal

```
DeviceProcessEvents
| where ProcessCommandLine has "advapi32" and ProcessCommandLine has "SetKernelObjectSecurity"
| join kind=inner DeviceFileEvents on DeviceName, FileName
| where FileName endswith ".everest"
```

(Note: `SetKernelObjectSecurity` invocation is rarely triggered from user-mode binaries; this is a high-confidence signal when correlated with `.everest` write events.)

7.5 Outbound WoL detection (NetFlow / Zeek)

```
udp.dst_port == 7 or udp.dst_port == 9
| filter packet_size between 102 and 144   # 102 = WoL payload; upper bound covers UDP/IP/
Ethernet framing variation across captors
| filter dst_ip in (subnet broadcast list)
| group by src_ip
| where count(distinct dst_mac) >= 3       # multiple wake targets in short window
```

The exact size depends on what your capture stack reports: 102 (Zeek `orig_bytes`, application payload), 110 (UDP datagram = payload + 8-byte UDP header), ≥130 (IP packet), ≥144 (full Ethernet frame). Calibrate the range against a known-good WoL appliance trace before deploying.

---

8. Sigma Rules

```
title: EVEREST Ransomware Pre-Encryption Setup Sequence
id: c79e0a1c-9b8d-4f1f-9b41-f3a2e1d8f0b1
status: experimental
description: |
  Detects the EVEREST ransomware pre-encryption configuration sequence based on
  observed CFA disable, SMB1 re-enable, firewall opening, and mountvol.exe usage
  occurring within a short window from the same parent process.
references:
  - https://www.ransomlook.io/analyses
author: ransomlook.io
date: 2026/05/07
logsource:
  product: windows
  category: process_creation
detection:
  selection_pwsh_cfa:
    Image|endswith: '\powershell.exe'
    CommandLine|contains|all:
      - 'Set-MpPreference'
      - 'EnableControlledFolderAccess'
      - 'Disabled'
  selection_pwsh_smb1:
    Image|endswith: '\powershell.exe'
    CommandLine|contains|all:
      - 'Enable-WindowsOptionalFeature'
      - 'SMB1Protocol'
  selection_netsh:
    Image|endswith: '\netsh.exe'
    CommandLine|contains|all:
      - 'advfirewall'
      - 'new enable=Yes'
  selection_mountvol:
    Image|endswith: '\mountvol.exe'
    CommandLine|contains: 'Volume{'
  selection_icacls:
    Image|endswith: '\icacls.exe'
    CommandLine|contains|all:
      - 'Everyone:F'
      - '/T /C /Q'
  condition: 3 of selection_*
falsepositives:
  - Server hardening exercises that intentionally enable SMB1 (rare and policy-controlled)
  - Penetration testing engagements
level: high
tags:
  - attack.impact
  - attack.t1486
  - attack.t1490
  - attack.t1489
  - attack.t1562.001

---
title: EVEREST Ransomware Wake-On-LAN Pre-Spread
id: e1f2c3d4-5a6b-7c8d-9e0f-1a2b3c4d5e6f
status: experimental
description: |
  Outbound UDP traffic to port 7 or 9 with a payload size consistent with a WoL
```

```
  magic packet (102-byte application payload), preceding a wave of SMB
  authentication or .everest file writes.
  Note: the literal 102-byte figure is the application-layer WoL payload
  (6 × 0xFF + 16 × MAC). Sensors may report sizes differently depending on
  whether they count payload-only (102), UDP datagram with header (110),
  IP packet (≥130), or full Ethernet frame (≥144). Tune the range to match
  the local capture stack.
logsource:
  product: zeek
  service: conn
detection:
  selection:
    proto: 'udp'
    resp_p:
      - 7
      - 9
    orig_bytes|gte: 102
    orig_bytes|lte: 144
  condition: selection
falsepositives:
  - Legitimate MDM/wake-on-LAN appliances (whitelist by source)
  - Network monitoring tools using UDP echo (port 7) — rare in modern environments
level: medium
tags:
  - attack.discovery
  - attack.t1018
```

9. Detection Confidence Matrix

For each signal, Confidence = how strongly the observation indicates this build line; Evasion difficulty = how hard it would be for the operator to remove or relocate the indicator without breaking core functionality.

Signal	Confidence	Evasion difficulty	Notes
Mutex `Global\7efc73f7-fda1-42d1-a4c5-8f1670bd08a5`	High (this build)	Low	Trivial to randomize the GUID per build
`.everest` file extension on encrypted output	High	Low	Trivial to rename — but breaks ransom-note matching
`EVERESTRANSOMWARE.txt` note filename	High	Low	Trivial to rename — but signals brand
3-of-N pre-encryption commands (CFA/SMB1/netsh/mountvol/icacls)	**High**	**High**	Operationally hard to remove without breaking spread/encrypt flow
`icacls "C:*" /grant Everyone:F /T /C /Q` (multi-drive)	High	Medium	Could be split into PowerShell `Set-Acl`, but the operational intent is detectable
Outbound UDP/7+9 broadcast, 102-byte payload	Medium	Medium	Distinctive in modern enterprise traffic; could be moved to TCP/RPC, but at engineering cost
WoL fan-out + ARP-derived target list	Medium	Medium	NDR/Zeek differentiator; survives most user-space evasions
`SetKernelObjectSecurity` from a user-mode binary writing `.everest`	High	High	Architectural to the DACL anti-tamper trick; removing it breaks self-protection
`HKCU\Software\AppName\PublicKey` Base64 blob (~684 chars)	High	Low	Trivial to relocate; high value as DFIR pivot if present
`[WALLNAME].bmp` literal in `%TEMP%`	Medium	Low	Build-template residue, easy to fix in next build
ConfuserEx DIE flags (Modified managed EP + Int confusion + Short names + Bad .cctor format)	Medium	Medium	Family-stable; ConfuserEx is widely shared, so confidence is build-level not family-level
3 worker threads with 4 s / 15 s / 2.5 s loop intervals	Medium	Medium	Distinctive cadence; fingerprint via memory analysis
Note typos `"trully"`, `"singifically"`, `"backups , etc. etc."`	Medium	Low	Cross-campaign signature, low cost to fix
Static PBKDF2 salt `01..08`	High	Low	Constant in `Program.Db`, easy to swap — but signals lazy crypto integration
`arp -a` parsed by a non-shell parent (managed binary)	Medium	High	Required for ARP-driven WoL; removing it breaks the spread chain
`mountvol.exe \\?\Volume{...}` from non-admin tooling context	High	Medium	Required to access unlettered volumes

10. Memory Artefacts (DFIR / live response)

Artefacts present in the running process's memory during or shortly after the encryption pass. These are valuable for live-response triage when the host is captured before self-delete completes.

Artefact	Location	Lifetime	DFIR value
AES-128 Key (16 bytes)	`Program.AesKey` static field	Process lifetime after first encryption call	Would permit decryption of files encrypted during the same process lifetime (Key+IV reused for all files in one run)
AES-128 IV (16 bytes)	`Program.AesIv` static field	Process lifetime	Pair with Key above
32-byte ASCII printable seed	Stack frame of `Helpers.GenSeed32Random`, then .NET string heap	~seconds (until GC collects)	Pre-PBKDF2 input — equivalent identifier to the Key+IV pair
RSA public key XML blob	Managed string heap (loaded by `FromXmlString`)	Process lifetime	Sample-pivot identifier (modulus extractable)
RSA-wrapped seed (Base64, ~684 chars)	Managed string heap, then written to registry	Process lifetime + persisted	Primary operator-side recovery token observed in this sample — also in `HKCU\Software\AppName\PublicKey` and the dropped note
Decoded UserStrings (note text, salutation, mutex GUID, blog URL, email, command lines)	Managed string heap, .NET string interning	Process lifetime	Cross-reference for cluster pivots and strings-based YARA
Mutex handle `Global\7efc73f7-fda1-42d1-a4c5-8f1670bd08a5`	Kernel object, handle in process table	Process lifetime	Sysmon Event 17 / handle table dump
Worker thread stacks (kill-loop bodies)	3 managed threads, no `Thread.Name` set (default worker IDs)	Process lifetime	Stack-walk identifies the 4 s / 15 s / 2.5 s loops; useful for in-memory YARA
Restart Manager session handle	Kernel object	Brief — closed after RmGetList iteration	`RmGetList` output enumerable from RM session ID if captured live
`Encryptor.TargetPathList`	Managed list field	Process lifetime	Full list of UNC + local paths queued for encryption — identifies victims if captured pre-encryption

Live-response priority: if the host can be paused or memory-imaged before `Program.Exit()`, recovering `Program.AesKey` + `Program.AesIv` would permit decryption of files encrypted during the same process lifetime, without the operator's RSA private key.

This recovery path is contingent on **all** of the following: - the locker process is still resident, **or** a coherent full-memory dump is captured before self-delete completes; - the static fields are not paged out (typical for active processes, but not guaranteed under memory pressure or VM-snapshot scenarios); - the dump tool produces a consistent view of the managed heap (live-acquisition smear can corrupt `byte[]` fields).

In practice: prefer a frozen-state acquisition (hibernation file, paused VM snapshot, ETW-suspended live dump) over a running-process dump where possible.

---

11. Defender Priority Queue

P0 (immediate)

1. **Isolate** the host showing the 3-thread fingerprint or any of the §5.1 commands.
2. **Block** outbound UDP/7 and UDP/9 broadcast traffic at the edge (rare in modern environments).
3. **Snapshot** memory of any host showing the mutex `Global\7efc73f7-fda1-42d1-a4c5-8f1670bd08a5`. Per §10, the AES Key+IV are derived in-process and held in `Program.AesKey` / `Program.AesIv` static fields during the encryption pass.
4. **Protect** the registry value `HKCU\Software\AppName\PublicKey` from being wiped — it contains the RSA-wrapped seed, which is the **primary operator-side recovery token observed in this sample** (a duplicate copy is also embedded in the dropped note).

P1 (≤1 h)

1. Scan domain for hosts with `EVERESTRANSOMWARE.txt` in `C:\ProgramData` (FIM signal).
2. Audit `LongPathsEnabled`, `LocalAccountTokenFilterPolicy`, `EnableLinkedConnections` registry values across the AD — the locker re-enables these. Any host with these set to `1` unexpectedly is a survivor or in-progress victim.
3. Verify SMB1 status on all servers (`Get-SmbServerConfiguration | select EnableSMB1Protocol`). Roll back if changed.
4. Pull Sysmon Event 17 for the mutex string across the SIEM window (mutex is created at the very first instructions of `Program.Main`).

P2 (≤24 h)

1. Hunt the `\[WALLNAME].bmp` literal in `%TEMP%` of all systems (the wallpaper artifact remains even after locker self-delete).
2. Reconstruct lateral movement: read `C:\ProgramData\<32-hex>` files (filename = MD5 of `Win32_Processor.ProcessorId` + `Win32_LogicalDisk.VolumeSerialNumber`). Each file contains the list of UNC paths added to `Encryptor.TargetPathList` for that host.
3. Audit and restore Defender CFA configuration.
4. Force credential rotation on impacted services (note text claims include "trusted representatives personal info" and "client risk levels" — claim is text-only but rotation is precautionary).

P3 (post-incident)

1. Cross-reference RSA-1024 modulus against shared Everest IOC corpora to confirm operator-side key reuse (typical: per-victim, but worth verifying).
2. The `System.Random` seed-recovery hypothesis (§2.3) is an analytical path, not an exploit pipeline. Pursuing it requires the boot timestamp to ms precision (Event Log `Microsoft-Windows-Kernel-General/12`) **and** the captured registry/note pair **and** demonstrated equivalence between reconstructed seed and intercepted ciphertext. Out of scope for this brief.

---

12. Observables Summary

The **Scope** column indicates the granularity at which an indicator is expected to be stable. Use this for hunting automation: filter by Family indicators for cross-campaign sweeps, Build indicators to cluster builds, Sample indicators to confirm a specific binary.

Type	Value	Scope	Confidence
SHA-256 (protected sample)	`1df92bf4c967297d8a39fc3f619a56702ee96d5cf9196b8e1d5b3654746c6514`	Sample	High
MD5 (protected sample)	`0dd70c334507188714bae0af7229b379`	Sample	High
SHA-256 (de4dot-cleaned x4)	`b30b7911b256daeceb631dec923f83c0ef17f34eb46452d5ab32aab2540be361`	Sample (analysis artefact, not a wild ITW indicator)	High
Mutex GUID `Global\7efc73f7-fda1-42d1-a4c5-8f1670bd08a5`	literal	Build (per-build variation expected)	High
Onion blog URL	`ransomocmou6mnbquqz44ewosbkjk3o5qjsl3orawojexfook2j7esad.onion`	Family (long-lived Everest blog)	High
Contact email	`everestaf@onionmail.org`	Family (Everest-stable pattern)	High
File extension	`.everest`	Family	High
Note filename	`EVERESTRANSOMWARE.txt`	Family	High
Note salutation prefix	`"Greetings from the Everest team"`	Family	High
Note typos	`"trully"`, `"singifically"`, `"backups , etc. etc."`	Family (cross-campaign signature)	High
RSA pubkey modulus (Base64)	`uByCMcH5MwQ1wXW3pKGb4eP8YVb3+d0Vrg243VI2f9jj00hM4CmRilFu+CrnNo/kZ/eMLOW13T/5RW1ono1c4uOtm6zDW/S229nc8eDjIEyqw9A6McuoVYYdW+lkm3u15pYG2uZBrv3MnAkQRllguCxnvvv+VT8K5rAHemyqVfc=`	Sample (per-victim pivot)	High
RSA pubkey exponent	`AQAB` (65537)	Family	High
Registry persistence path	`HKCU\Software\AppName\PublicKey`	Build	High
Wallpaper template literal	`%TEMP%\[WALLNAME].bmp`	Build (template residue)	High
Spread-tracking file path	`C:\ProgramData\<MD5(ProcessorId+VolumeSerial)>` (no extension)	Build (algorithm stable)	High
User-Agent (dead code, unreachable)	`Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 ... Edge/17.17134`	Sample	Low
WoL UDP ports	7, 9 (broadcast, 102-byte payload)	Family (across .NET Everest builds)	Medium
WoL hardcoded subnet mask	`255.255.255.0`	Family	Medium
ConfuserEx DIE flags	`Modified managed EP + Int confusion + Short names + Bad .cctor format`	Build	Medium

Type	Value	Scope	Confidence
Static PBKDF2 salt	`01 02 03 04 05 06 07 08`	Family (build-stable across known builds)	High
Worker thread cadence	4 s / 15 s / 2.5 s	Family	Medium
String decryptor signature	`<Module>.m(string)` — GZip + Base64 + lookup	Family	High

13. Conclusion

Observed facts about this sample:

- Self-identified as Everest via embedded strings (`.everest` extension, `EVERESTRANSOMWARE.txt` note filename, `"Greetings from the Everest team"` salutation, `everestaf@onionmail.org` contact, blog onion URL).
- ConfuserEx 1.x (or compatible fork) protection with watermark stripped; runtime string decryption via `<Module>.m(string)` (GZip+Base64+lookup).
- Crypto stack as deployed: AES-128-CBC PKCS#7 (small files) and AES-128-CBC NoPadding chunked (>10 MB), key+IV from PBKDF2-HMACSHA1 with static 8-byte salt and 1000 iterations, seed from `System.Random.Next(33,127)` for 32 bytes; the seed is wrapped with RSA-1024 PKCS#1 v1.5 using a sample-embedded public key.
- Three concurrent worker threads: RE-tool kill (4 s), service+process kill (15 s), memory-pig kill at >250 MB threshold (2.5 s).
- Pre-encryption operations include CFA disable on Windows 10/11, SMB1 enable, `LongPathsEnabled=1`, `LocalAccountTokenFilterPolicy=1`, `EnableLinkedConnections=1`, network-discovery services started, `icacls` grants Everyone full control on C:/D:/Z:, `mountvol.exe` enumerates and mounts `\\?\Volume{...}`.
- LAN spread: ARP cache parsing → Wake-On-LAN UDP/7 + UDP/9 broadcast; share enumeration via `net view`, `NetDfsEnum`, `WNetEnumResource`, WMI `Win32_Share`, `Win32_NetworkConnection`, `Win32_MappedLogicalDisk`; `net use \\<unc>` per discovered share; UNC list persisted to `C:\ProgramData\<MD5(ProcessorId+VolumeSerial)>`.
- Process self-protection: own DACL is mutated with an AccessDenied ACE for the World SID — defeats standard user-mode termination paths.
- Anti-Raccine: deletion of IFEO entries, `HKLM\SOFTWARE\Raccine`, `HKCU\…\Run\Raccine Tray`, scheduled task `Raccine Rules Updater`, plus `taskkill` of `Raccine.exe` / `RaccineSettings.exe`.
- Recovery inhibition: `vssadmin Delete Shadows /all /quiet` (or `Get-CimInstance Win32_ShadowCopy | Remove-CimInstance`), `Srclient!SRRemoveRestorePoint` for every restore point, `del /s /f /q` against backup file patterns on drives c-h.
- The note text contains the claim of ~1 TB exfiltration. **No exfiltration code is present in the binary.**

Inferred: the build line is consistent with observed Everest tradecraft. Family-level attribution is supported by the embedded strings; broader operator-cluster attribution is out of scope without corpus comparison.

Sample-unique items (not portable to other builds): the RSA-1024 modulus in `<Module>.smethod_19` and the salutation string in `<Module>.smethod_5`. All other operational strings, the mutex GUID, the contact email and the onion URL appear in this sample as decoded literals; their portability across other Everest builds is not determined within this analysis.