Payload

Technical Analysis — Esxi

REVERSE-ENGINEERED REPORT

RansomLook · ransomlook.io
File last modified: 2026-04-26

Sample SHA-256: `bed8d1752a12e5681412efbb8283910857f7c5c431c2d73f9bbc5b379047a316`

Payload Ransomware (ESXi)

1. Sample Identification

Field	Value
SHA-256	`bed8d1752a12e5681412efbb8283910857f7c5c431c2d73f9bbc5b379047a316`
Type	ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked
Interpreter	`/lib64/ld-linux-x86-64.so.2`
Build	for GNU/Linux 2.6.32, stripped
Build ID	`04279f213ecf1f20b81fcf4dc3a0e87abb7f412b`
Size	39 904 bytes (39 KB)
Language	C
Target	VMware ESXi hypervisors
Functions	180 total (137 named, 43 unnamed)
Strings	98

Dynamic Libraries

Library	Functions Used
libxml2.so.2	`xmlReadFile`, `xmlXPathNewContext`, `xmlXPathEvalExpression`, `xmlNodeGetContent`, `xmlStrcmp`, `xmlXPathFreeObject`, `xmlXPathFreeContext`, `xmlFreeDoc`, `xmlFree`, `xmlCleanupParser`
libpthread.so.0	`pthread_create`, `pthread_detach`, `pthread_mutex_init/lock/unlock`, `pthread_cond_init/wait/signal/broadcast`, `sigaction`
libc.so.6	Standard file I/O, memory, string functions

2. Anti-Analysis

Anti-Debug — `anti_debug_check_tracerpid` @ `0x40622F`

Reads `/proc/self/status` line by line, searches for `TracerPid:` field. If value != 0 (debugger attached), calls `exit_if_debugged` @ `0x4062AD` which terminates the process.

String Obfuscation — RC4 with key "FBI"

All sensitive strings are RC4-encrypted in the `.data` section. The decryption function `rc4_decrypt_string` @ `0x402229`: 1. Initializes RC4 KSA (`rc4_ksa` @ `0x406189`) with the first 3 bytes of `"FBIthread-pool-%d"` → key = `FBI` 2. Applies RC4 PRGA XOR (`rc4_prga_xor` @ `0x4061DF`) to decrypt the string 3. Returns pointer to global buffer `byte_609460`

Instance Lock — `check_instance_lock` @ `0x4025D9`

Uses `flock()` to ensure single instance execution. If another instance is running, exits.

3. Decrypted Strings

Address	Length	Decrypted Value	Purpose
`0x6092A0`	33	`/etc/vmware/hostd/vmInventory.xml`	VMware VM inventory file
`0x609288`	13	`//ConfigEntry`	XPath query to find VM entries
`0x60925A`	5	`objID`	XML field: VM numeric ID
`0x609250`	10	`vmxCfgPath`	XML field: path to VM's .vmx config
`0x609220`	44	`/usr/lib/vmware/hostd/docroot/ui/welcome.txt`	Ransom note file path (ESXi web UI)
`0x609260`	14	`no ConfigEntry`	Error: no VMs found
`0x609270`	18	`xmlReadFile failed`	Error: can't read inventory
`0x6093F0`	18	`set_key(): failed!`	Error: crypto init failure
`0x6092D0`	30	(format string for note write error)	Error writing note

---

4. Execution Flow

`cicada_main` @ `0x401980` (main function, 0x44A bytes)

```
1. anti_debug_check_tracerpid()         → exit if debugged
2. check_instance_lock()                → exit if already running
3. init_ransom_note()                   → decrypt & write ransom note
4. cpuid_check_avx()                    → select crypto implementation
   - bit 2 (AVX2): use ChaCha20 (sub_4032E6)
   - bit 0 (SSE2): use Salsa20 (sub_402F85)
5. thpool_init(2 * sysconf(_SC_NPROCESSORS_ONLN))
6. Parse -i argument: comma-separated VM IDs to skip
7. RC4 decrypt "/etc/vmware/hostd/vmInventory.xml"
8. xmlReadFile(inventory_path)
9. xmlXPathEvalExpression("//ConfigEntry")
10. For each ConfigEntry:
    a. Extract "objID" → VM numeric ID
    b. Extract "vmxCfgPath" → path to VM's .vmx file
    c. Skip if ID is in -i exclusion list
    d. Extract directory from vmxCfgPath (parent dir = VM datastore)
    e. walk_directory_encrypt(vm_directory)
11. Cleanup: free XML, destroy thread pool
```

VMware Inventory Parsing

The file `/etc/vmware/hostd/vmInventory.xml` is the ESXi VM inventory. Each `<ConfigEntry>` contains: - `<objID>` : numeric VM identifier - `<vmxCfgPath>` : full path to the VM's `.vmx` configuration file (e.g., `/vmfs/volumes/datastore1/vm1/vm1.vmx`)

The ransomware extracts the parent directory of the `.vmx` file to find the VM's datastore directory, then recursively encrypts all files in it.

---

5. Encryption System

Architecture

```
walk_directory_encrypt(dir_path)
├── opendir / readdir64
├── For each regular file:
│   ├── Skip if already has extension (.xx0001)
│   └── thpool_add_work(encrypt_file, filepath)
└── closedir

encrypt_file(filepath)
├── fopen64(filepath, "rb+")
├── flock(fd, LOCK_EX)
├── Check filesize > 5 GB (0x140000000) — skip smaller files
├── generate_random_32bytes() → ephemeral private key (clamped for Curve25519)
├── generate_random_32bytes() → nonce
├── x25519_dh(ephemeral_priv, note_pubkey) → shared secret for note
├── x25519_dh(ephemeral_priv, master_pubkey) → shared secret for encryption
├── Init stream cipher (Salsa20 or ChaCha20) with shared secret + nonce
├── Split file into 5 equal parts
│   └── For each part: encrypt 1 GB (0x40000000) in 1 MB chunks
│       ├── fseeko64 to offset
│       ├── fread → cipher XOR → fwrite (in-place)
│       └── Advance to next 1/5th of file
├── Write "payload" marker (7 bytes)
├── RC4-encrypt metadata (56 bytes: ephemeral pubkey + nonce)
├── Append encrypted metadata to end of file
├── fflush + fclose
└── rename(filepath, filepath + ".xx0001")
```

Crypto Primitives

Primitive	Function	Address
X25519 (Curve25519 ECDH)	`x25519_dh`	`0x405359`
Curve25519 scalar multiplication	`curve25519_scalarmult`	`0x405076`
Salsa20 (SSE2 optimized)	`salsa20_encrypt_block`	`0x402F85`
ChaCha20 (AVX2 optimized)	`chacha20_encrypt_block`	`0x4032E6`
RC4 (KSA)	`rc4_ksa`	`0x406189`
RC4 (PRGA XOR)	`rc4_prga_xor`	`0x4061DF`
CSPRNG	`generate_random_32bytes` → `read_urandom`	`0x402EC0` → `0x402F04`

Key Material

Two base64-encoded public keys are embedded in `.rodata` : - **Master public key** (`off_609428`): used for X25519 DH to derive the file encryption key - **Note public key** (`off_609420`): used for X25519 DH to derive a secondary shared secret (stored in metadata)

The Salsa20/ChaCha20 constant `"expand 32-byte k"` is present at `0x406398` .

Encryption Strategy

- **Threshold**: only files > 5 GB are encrypted — specifically targets VMDK virtual disk files
- **5-pass partial encryption**: the file is divided into 5 equal segments. For each segment, 1 GB is encrypted from the start of that segment
- **Result**: ~5 GB encrypted out of potentially 100+ GB, ensuring the VMDK is unbootable while keeping encryption fast
- **Per-file keys**: each file gets a unique ephemeral X25519 keypair → unique stream cipher key
- **In-place**: data is read, XORed with keystream, and written back at the same offset

File Metadata (56 bytes)

Appended at the end of each encrypted file:

```
[original file data (partially encrypted)]
["payload" marker — 7 bytes]
[RC4-encrypted metadata — 56 bytes: ephemeral X25519 public key (32B) + nonce (24B)]
```

The metadata is RC4-encrypted using the key derived from `"FBIthread-pool-%d"` (3 bytes: `FBI`).

---

6. Ransom Note

Deployment

The ransom note is deployed to `/usr/lib/vmware/hostd/docroot/ui/welcome.txt` — this is the **ESXi web UI welcome page**, meaning administrators will see the note when accessing the ESXi management interface via browser.

The note content is stored base64-encoded and RC4-encrypted in `.rodata`: - **Encrypted blob**: `0x4063C3` (2528 bytes base64 → 1896 bytes encrypted) - **RC4 key**: `0x4083C4` (base64 → 32 bytes: `NrjSayDQXDJOMDdPlyCsLzm26VJ3K5jK`)

Full Ransom Note

```
Welcome to Payload!

The next 72 hours will determine certain factors in the life of your company:
the publication of the file tree, which we have done safely and unnoticed
by all of you, and the publication of your company's full name on our
luxurious blog.
NONE of this will happen if you contact us within this time frame and our
negotiations are favorable.

We are giving you 240 hours to:
1. familiarize yourself with our terms and conditions,
2. begin negotiations with us,
3. and successfully conclude them.
The timer may be extended if we deem it necessary (only in the upward direction).
Once the timer expires, all your information will be posted on our blog.

ATTENTION!
Contacting authorities, recovery agencies, etc. WILL NOT HELP YOU!
At best, you will waste your money and lose some of your files, which they
will carefully take to restore!
You should also NOT turn off, restart, or put your computer to sleep.
In the future, such mistakes can make the situation more expensive and the
files will not be restored!
We DO NOT recommend doing anything with the files, as this will make it
difficult to recover them later!

When contacting us:
you can request up to 3 files from the file tree,
you can request up to 3 encrypted files up to 15 megabytes
so that we can decrypt them and you understand that we can do it.

First, you should install Tor Browser:
1. Open: https://www.torproject.org/download
2. Choose your OS and select it
3. Run installer
4. Enjoy!

In countries where tor is prohibited, we recommend using bridges,
which you can take: https://bridges.torproject.org/

You can read:
http://payloadrz5yw227brtbvdqpnlhq3rdcdekdnn3rgucbcdeawq2v6vuyd.onion (Tor)

To start negotiations, go to:
http://payloadynyvabjacbun4uwhmxc7yvdzorycslzmnleguxjn7glahsvqd.onion

User: [SNIP]
Password: [SNIP]

Your ID to verify: [SNIP]
```

7. IOCs

Hashes

Type	Value
SHA-256	`bed8d1752a12e5681412efbb8283910857f7c5c431c2d73f9bbc5b379047a316`
MD5	`f91cbdd91e2daab31b715ce3501f5ea0`
Build ID	`04279f213ecf1f20b81fcf4dc3a0e87abb7f412b`

Network

Type	Value
DLS Onion	`payloadrz5yw227brtbvdqpnlhq3rdcdekdnn3rgucbcdeawq2v6vuyd.onion`
Chat Onion	`payloadynyvabjacbun4uwhmxc7yvdzorycslzmnleguxjn7glahsvqd.onion`

Files

Path	Description
`/usr/lib/vmware/hostd/docroot/ui/welcome.txt`	Ransom note (ESXi web UI)
`/etc/vmware/hostd/vmInventory.xml`	Parsed for VM paths
`*.xx0001`	Encrypted files

Extension

`.xx0001`

Distinctive Strings

- `"FBIthread-pool-%d"` (RC4 key source + thread pool naming)
- `"payload"` (marker written before file metadata)
- `"expand 32-byte k"` (Salsa20/ChaCha20 constant)
- `"/proc/self/status"` + `"TracerPid:"` (anti-debug)
- `".xx0001"` (extension in .rodata)

---

8. MITRE ATT&CK Techniques

ID	Technique	Implementation
T1486	Data Encrypted for Impact	X25519 + Salsa20/ChaCha20, .xx0001 extension, partial 5-pass encryption
T1082	System Information Discovery	Parses `/etc/vmware/hostd/vmInventory.xml` for VM inventory
T1083	File and Directory Discovery	Recursive directory walk, file size filtering (>5 GB)
T1027	Obfuscated Files or Information	RC4-encrypted strings (key "FBI"), base64-encoded note
T1497.001	Virtualization/Sandbox Evasion: System Checks	Anti-debug via TracerPid
T1480	Execution Guardrails	Instance lock via flock, VM ID exclusion via -i argument
T1491.001	Internal Defacement	Overwrites ESXi web UI welcome page
T1059.004	Command and Scripting Interpreter: Unix Shell	Uses `system()` (linked but usage TBD)