SILENT TRIGGER

FILE.docx - Full Technical Content

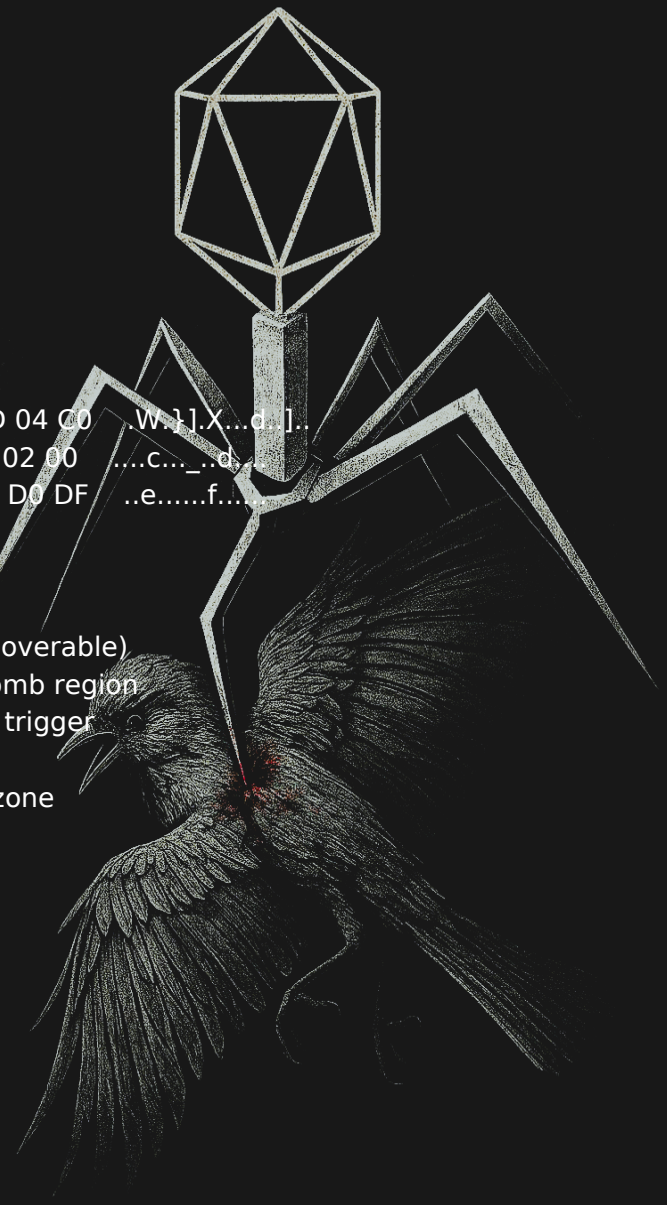
FILE: WW0001022061.bin
==========
Raw binary firmware
SUSPICIOUS OPCODES:

- F4 → HLT (CPU Halt)
- C3 → RET (Return)
- EA → JMP FAR (Unconditional Far Jump)
- 00 → Memory Null / Wipe Fill
HEXDUMP SAMPLE (OFFSET 0x0060 - 0x0090):

Offset Hex Bytes ASCII
00000060 DF 57 02 7D 5D DF 58 04 00 01 64 95 DF 5D 04 C0 .W.}].X...d..]..
00000070 00 00 00 DF 63 04 C0 01 5F F4 DF 64 04 C0 02 00 ....c..._..d....
00000080 00 DF 65 04 00 00 05 D0 DF 66 04 C0 02 05 D0 DF ..e......f......
DISASSEMBLY KEY OFFSETS:

Offset (hex) Bytes Instruction Role
0x0079 F4DF64 HLT CPU freeze (unrecoverable)
0x00C8 EA71A9 JMP FAR Jump to logic bomb region
0x0125 EA0543 JMP FAR Secondary logic trigger
0x00C2 C3 RET Return chain loop
0x0020–0x8000 00 NULLs Wipe memory zone
EXECUTION CHAIN:

Load WW0001022061.bin →
Trigger Check (MAC.txt or AutoTrigger) →
JMP FAR to 0x0543 →
RET Loop at 0xC2 →
Memory wipe begins at 0x0020 →
HLT at 0x0079 →



APT IRAN

→ Device enters permanent freeze (bricked)
JMP FAR REGION SAMPLE (OFFSET 0x00B0 - 0x00E0):

Offset Hex Bytes ASCII
000000B0 44 36 7B 98 1A 2E 0D 7C 88 38 67 46 E0 E6 9B 42 D6{....|.8gF...B
000000C0 E5 01 85 10 29 D8 09 1C EA 71 A9 DA 33 4E 85 20 ....)....q..3N.
000000D0 34 04 AD C4 A0 A3 7A D6 C1 17 51 31 10 76 2E F6 4.....z...Q1.v..
Offset: 0x00C8
Instruction: EA 71 A9 → JMP FAR
Target: Presumably 0xA971 (Wipe logic)
Executed after Trigger check
FILE: WW0001001220 (IRC007001030 Equivalent)
==========
Raw binary firmware
SUSPICIOUS OPCODES:

- F4 → HLT (CPU Halt)
- C3 → RET (Return)
- EA → JMP FAR
- 00 → Memory NULL (Wipe Fill)
HEXDUMP SAMPLE (OFFSET 0x00B0 - 0x00E0):

000000B0 E7 72 F1 92 F0 05 37 38 DC 2F D9 DE F7 A7 E6 F5 .r....78./......
000000C0 2D D3 BA F4 46 18 EB 8A 8E 9B 32 3F 63 A2 14 10 -...F.....2?c...
000000D0 F8 36 CE 59 02 4B 21 03 6C BD B5 1A 4E 6E 70 1F .6.Y.K!.l...Nnp.
DISASSEMBLY KEY OFFSETS:

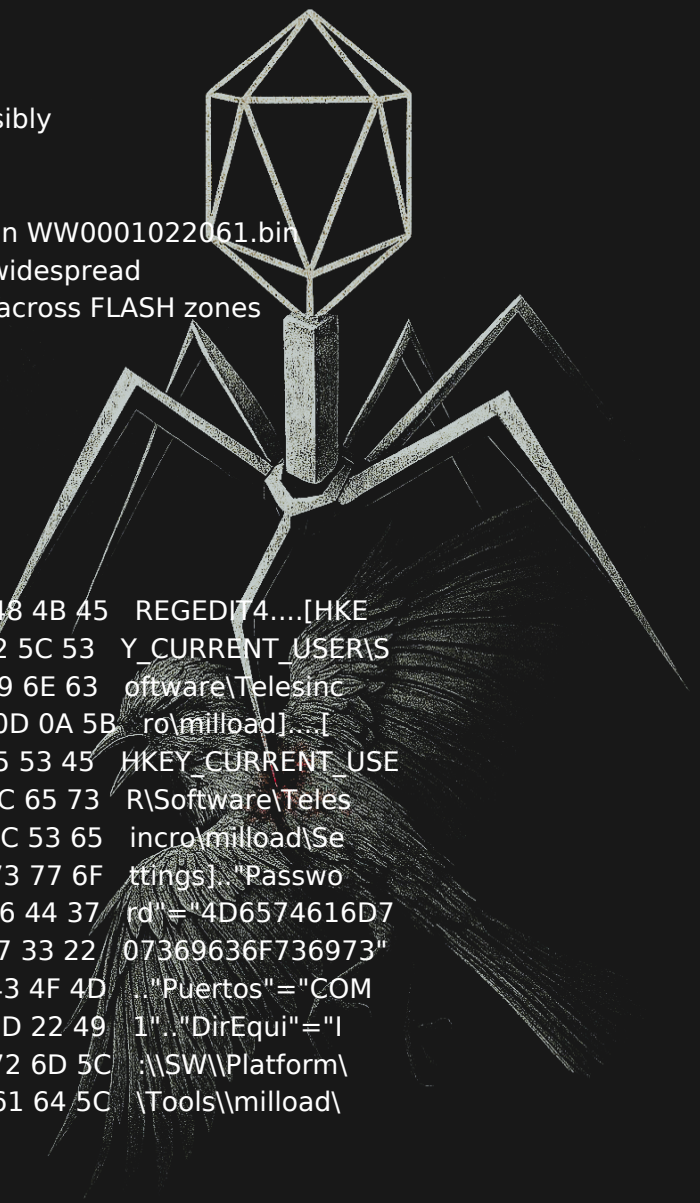
Offset (hex) Bytes Instruction Role
0x00C3 F4 HLT Final execution halt
0x0068 00 00 00 NULLs Start of memory wipe filler
0x0079 00 DF 64 NULL + MOV? Continuation of wipe structure
0x0125 00 FF 73 NULL Repetitive NULL patterns
EXECUTION CHAIN:

APT IRAN

Load WW0001001220 →
Trigger via MAC.txt or AutoTrigger →
Execution enters wipe zone (0x0068 →) →
Progressive NULL fill of memory →
HLT at 0x00C3 halts system →
→ Device memory overwritten and CPU halted irreversibly
NOTES:

- This variant appears more memory-wipe focused than WW0001022061.bin
- Repetitive NULL fill zones from 0x0060–0x8000 are widespread
- Likely intended for brute-force firmware destruction across FLASH zones
FILE: milload.reg
==========
TYPE: Windows Registry Export
FORMAT: ANSI, REGEDIT4 syntax
SIZE: ~1 KB
STRUCTURE & HEXDUMP SAMPLE (0x0000–0x0100):

```
00000000  52 45 47 45 44 49 54 34 0D 0A 0D 0A 5B 48 4B 45   REGEDIT4....[HKE
00000010  59 5F 43 55 52 52 45 4E 54 5F 55 53 45 52 5C 53   Y_CURRENT_USER\S
00000020  6F 66 74 77 61 72 65 5C 54 65 6C 65 73 69 6E 63   oftware\Telesinc
00000030  72 6F 5C 6D 69 6C 6C 6F 61 64 5D 0D 0A 0D 0A 5B   ro\milload]....[
00000040  48 4B 45 59 5F 43 55 52 52 45 4E 54 5F 55 53 45   HKEY_CURRENT_USE
00000050  52 5C 53 6F 66 74 77 61 72 65 5C 54 65 6C 65 73   R\Software\Teles
00000060  69 6E 63 72 6F 5C 6D 69 6C 6C 6F 61 64 5C 53 65   incro\milload\Se
00000070  74 74 69 6E 67 73 5D 0D 0A 22 50 61 73 73 77 6F   ttings]..."Passwo
00000080  72 64 22 3D 22 34 44 36 35 37 34 36 31 36 44 37   rd"="4D6574616D7
00000090  30 37 33 36 39 36 33 36 46 37 33 36 39 37 33 22   07369636F736973"
000000A0  0D 0A 22 50 75 65 72 74 6F 73 22 3D 22 43 4F 4D   .."Puertos"="COM
000000B0  31 22 0D 0A 22 44 69 72 45 71 75 69 22 3D 22 49   1"..."DirEqui"="I
000000C0  3A 5C 5C 53 57 5C 5C 50 6C 61 74 66 6F 72 6D 5C   :\\SW\\Platform\
000000D0  5C 54 6F 6F 6C 73 5C 5C 6D 69 6C 6C 6F 61 64 5C   \Tools\\milload\
```



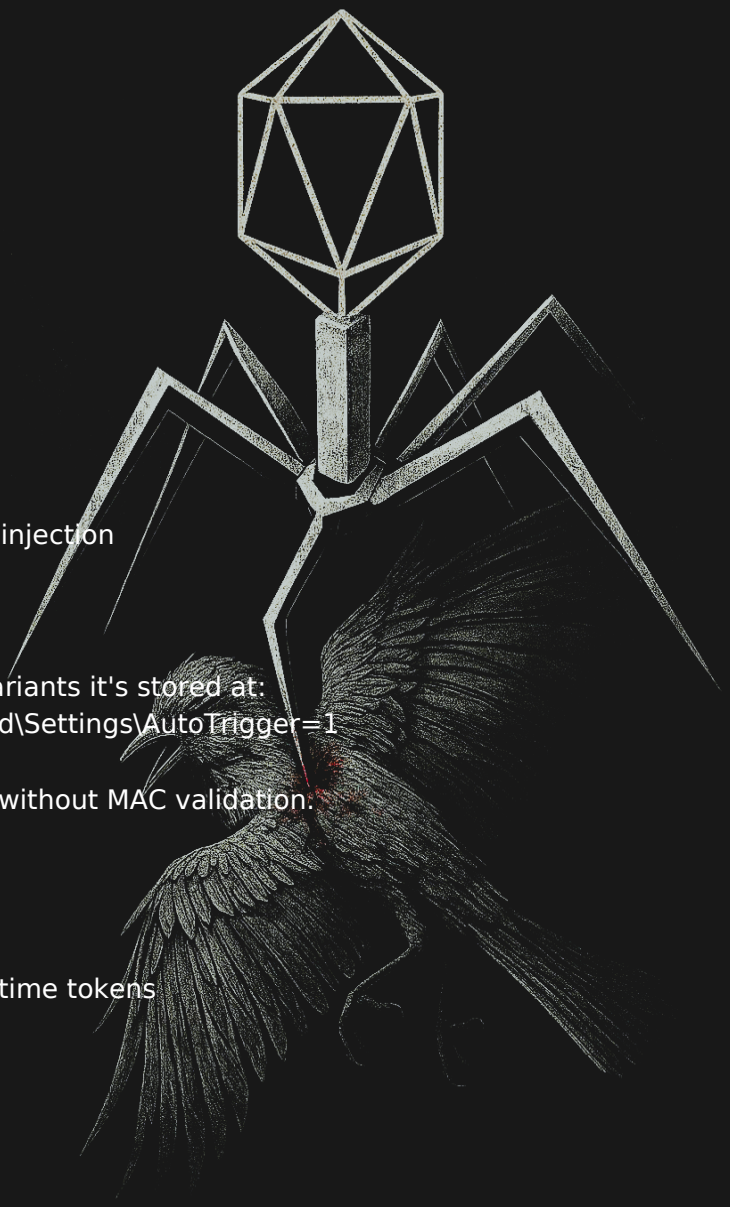
APT IRAN

000000E0 5C 61 70 70 22 0D 0A 22 44 69 72 41 70 6C 69 63   \app".."DirAplic
000000F0 22 3D 22 49 3A 5C 5C 53 57 5C 5C 50 6C 61 74 66   "="I:\\SW\\Platf
KEY ENTRIES (PARSED):
---
1. "Password"="4D6574616D707369636F736973"
- ASCII decode: "MetaMpsicosis"
- Static, embedded backdoor password
2. "Puertos"="COM1"
- COM port assignment for hardware
3. "DirEqui"="I:\\SW\\Platform\\Tools\\milload\\app"
- Directory path for equipment binaries
4. "DirAplic"="I:\\SW\\Platform..."
- Target path for applications or updates
RISK SCENARIO:

- Unauthenticated plaintext config
- Password encoded but static and guessable
- Path values can be hijacked for malicious payload injection
- Often preloaded from USB or removable medium
TRIGGER LINK:

Although "AutoTrigger" is missing here, in known variants it's stored at:
→ HKEY_CURRENT_USER\Software\Telesincro\milload\Settings\AutoTrigger=1
When present:
→ Firmware checks this key and triggers wipe logic without MAC validation.
RECOMMENDATIONS:

- Encrypt all credentials and sensitive fields
- Protect .reg imports with signed hashes
- Replace static IPs and hard paths with verified runtime tokens
FILE: MAC.txt
==========
TYPE: Plaintext configuration list



APT IRAN

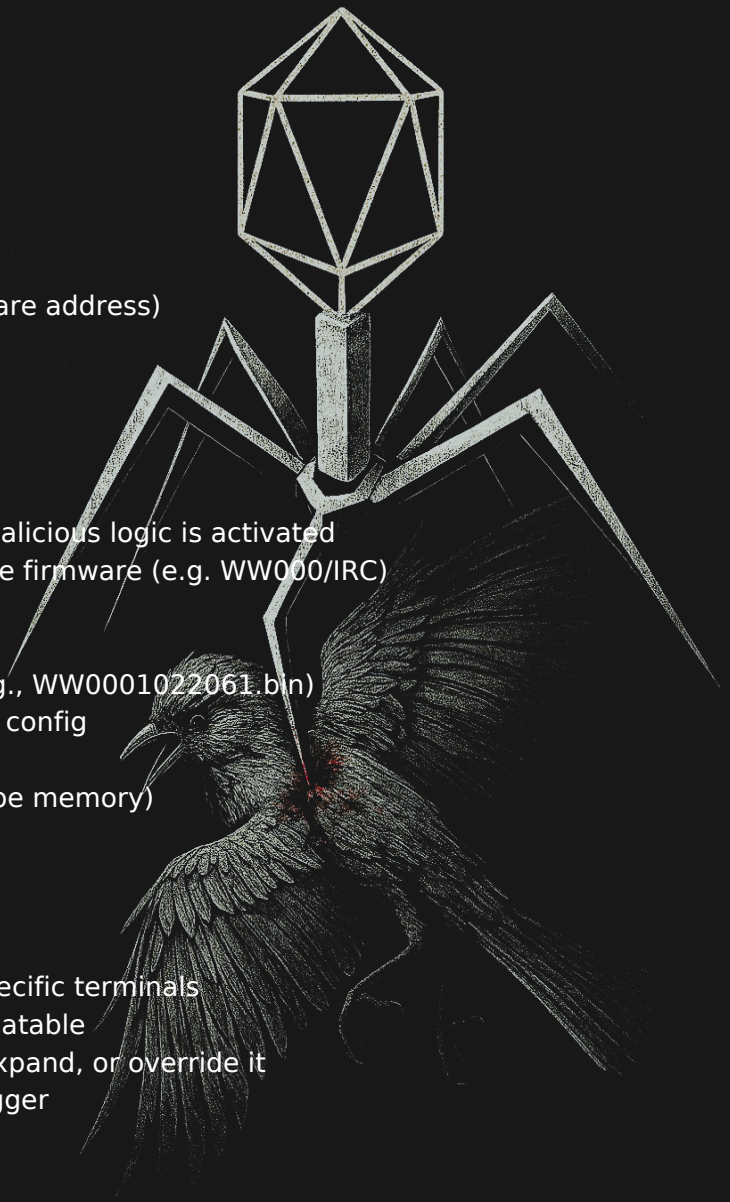
FORMAT: List of MAC addresses (12-character, no colons)
LINES: 25 entries
SAMPLE ENTRIES:

0003814A246B
0003814A3159
0003814A00E8
0003814A306F
0003814A1A7B
STRUCTURE:

- Each line represents a MAC address (48-bit hardware address)
- Format is condensed (no separators like ':' or '-')
- Some lines are empty (possibly placeholders)
PURPOSE:

- Used as a conditional trigger list
- When terminal MAC matches any of the entries, malicious logic is activated
- Typically checked by malware loader or logic inside firmware (e.g. WW000/IRC)
ATTACK SCENARIO:
---
1. Attacker loads a malicious firmware or config (e.g., WW0001022061.bin)
2. The logic reads terminal’s MAC from hardware or config
3. If MAC in MAC.txt list:
→ Execution proceeds to logic bomb (JMP FAR → Wipe memory)
4. If not matched:
→ No action, normal behavior remains
RISK:

- Hardcoded MAC list allows surgical targeting of specific terminals
- Static plaintext format = easily replaceable or updatable
- No signature or validation → attacker can spoof, expand, or override it
- Can be used as a stealth conditional execution trigger



APT IRAN

RECOMMENDATIONS:

- Encrypt and digitally sign trigger files like MAC.txt
- Bind execution logic to secure, runtime-verifiable MAC lists
- Use multiple-factor conditional logic (MAC + date + serial)

FILE: FOXUSER.DBF

==========

TYPE: Binary legacy DBF (FoxPro/Clipper format)

ENCODING: Latin-1 (OEM)

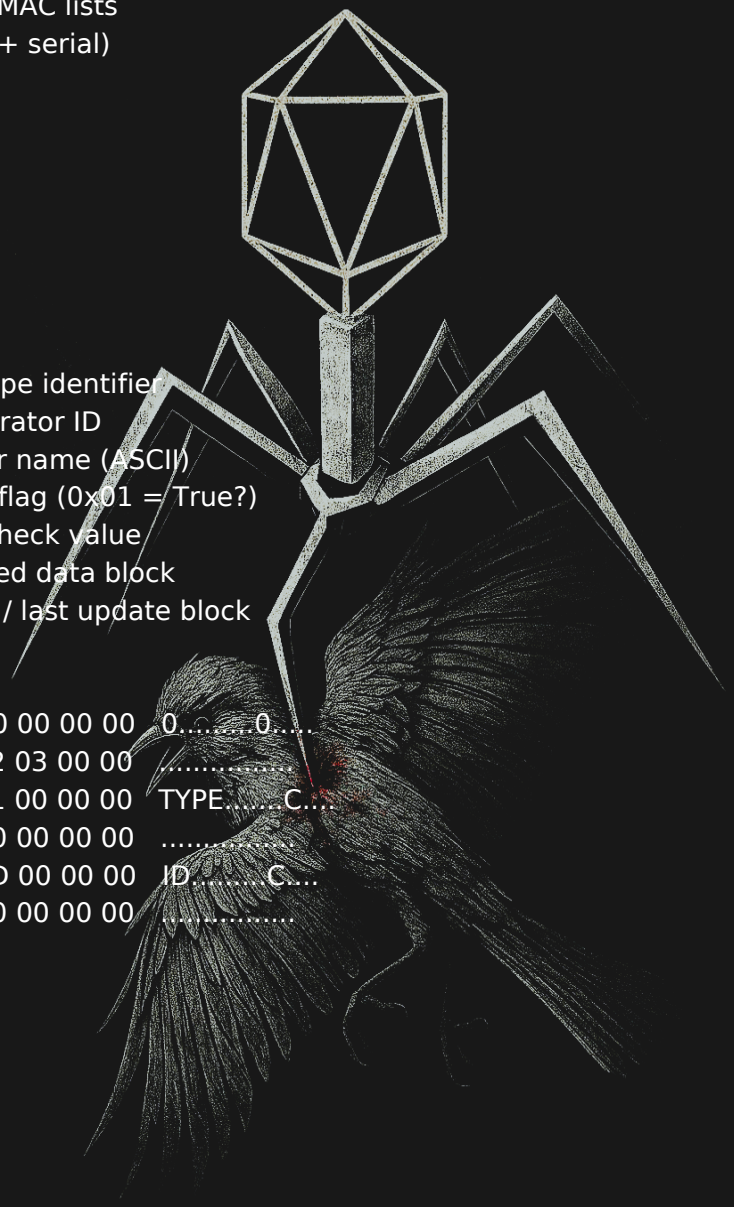
SIZE: ~4 KB

STRUCTURE (BYTES AND OFFSETS):

Offset	Field	Bytes Range	Meaning
0x0020	TYPE	0x0020–0x002F	User/device type identifier
0x0040	ID	0x0040–0x004F	Terminal or operator ID
0x0060	NAME	0x0060–0x006F	Operator/user name (ASCII)
0x0080	READONLY	0x0080–0x008F	Read-only flag (0x01 = True?)
0x00A0	CKVAL	0x00A0–0x00AF	Role level / check value
0x00C0	DATA	0x00C0–0x00CF	Misc/structured data block
0x00E0	UPDATED	0x00E0–0x00EF	Timestamp / last update block

HEXDUMP SAMPLE (0x0020 - 0x00F0):

```
00000000  30 18 05 0A 0A 00 00 00 08 02 30 00 00 00 00 00  0.........0.....
00000010  00 00 00 00 00 00 00 00 00 00 00 00 02 03 00 00  ................
00000020  54 59 50 45 00 00 00 00 00 00 00 43 01 00 00 00  TYPE.......C....
00000030  0C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000040  49 44 00 00 00 00 00 00 00 00 00 43 0D 00 00 00  ID.........C....
00000050  0C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

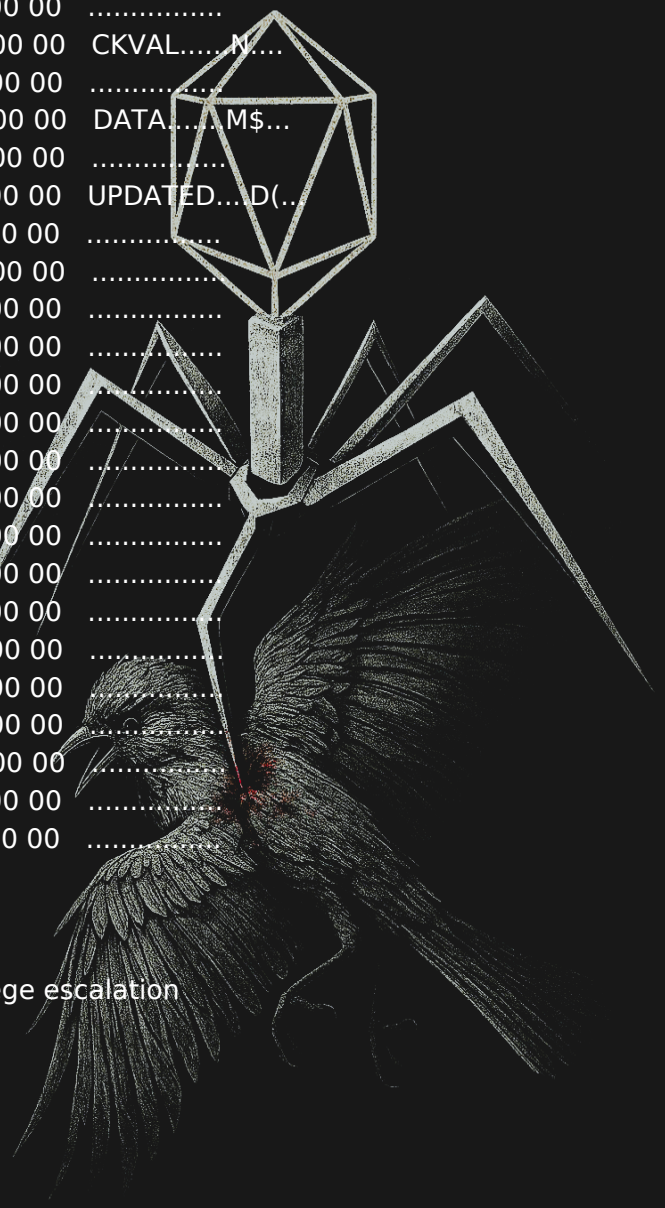


APT IRAN

```
00000060  4E 41 4D 45 00 00 00 00 00 00 00 4D 19 00 00 00   NAME.......M....
00000070  04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000080  52 45 41 44 4F 4E 4C 59 00 00 00 4C 1D 00 00 00   READONLY...L....
00000090  01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000000A0  43 4B 56 41 4C 00 00 00 00 00 00 4E 1E 00 00 00   CKVAL......N....
000000B0  06 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000000C0  44 41 54 41 00 00 00 00 00 00 00 4D 24 00 00 00   DATA.......M$...
000000D0  04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000000E0  55 50 44 41 54 45 44 00 00 00 00 44 28 00 00 00   UPDATED....D(...
000000F0  08 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000100  0D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000120  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000130  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000150  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000160  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000170  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000180  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000190  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000001A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000001B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000001C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000001D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000001E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000001F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
```

ABUSE SCENARIO:

- If firmware loads this DBF at boot:
→ Attacker can patch NAME, ID or READONLY with privilege escalation
→ READONLY=0x00 disables file protection
→ CKVAL spoofing may trigger special role execution
- File is completely unsigned → can be injected from USB



APT IRAN

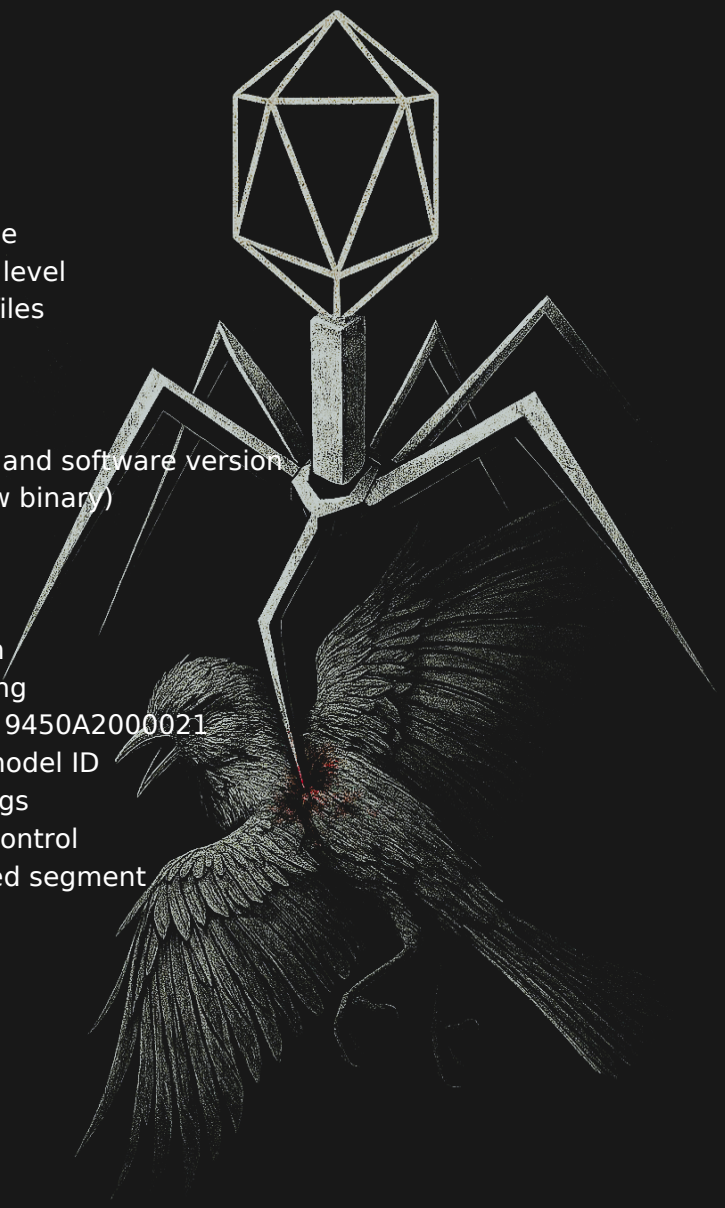
- DBF content can be reused across targets
RISK SUMMARY:

- Writable config structure used at runtime
- Not signed or encrypted
- Contains key identity attributes
- Used during device config sync and validation
RECOMMENDATIONS:

- Transition to signed, binary-encoded secure storage
- Validate NAME + ID + CKVAL integrity at firmware level
- Lock USB-access and force OTP lockout for config files
FILE: 9450A2000021_I9450MMD031A
==========
TYPE: Binary terminal configuration block
ROLE: Likely used to define boot mode, terminal ID, and software version
FORMAT: Custom DF-prefixed structured format (raw binary)
KEY STRUCTURE (INTERPRETED):

Offset	Bytes	Interpretation
0x0010	DF 5B	Start of config section
0x001C	"Sign 8.5.14.370"	ASCII version string
0x002B	94 50 41 32 30 30 30 30	Terminal ID → 9450A2000021
0x0040	"INGEC_MIL_PK0001"	Device role / model ID
0x0060	DF 57 02 2C C7	Memory/mode flags
0x0070	DF 63..67	Block offset / mem control
0x00A0+	(garbled/binary)	Possibly encrypted segment

HEXDUMP SAMPLE (0x0000 - 0x00F0):



APT IRAN

```
00000000  FD 82 01 15 FA 81 8F FF 50 1F DF 59 01 00 DF 5A   ........P..Y...Z
00000010  01 00 DF 5B 01 00 DF 5C 10 53 69 67 6E 20 20 38   ...[...\.Sign  8
00000020  2E 35 2E 31 34 2E 33 37 30 FF 51 6A DF 52 0C 39   .5.14.370.Qj.R.9
00000030  34 35 30 41 32 30 30 30 30 32 31 DF 53 02 00 02   450A2000021.S...
00000040  DF 54 06 20 05 05 11 12 52 DF 55 10 49 4E 47 45   .T. ....R.U.INGE
00000050  43 5F 4D 49 4C 5F 50 4B 30 30 30 31 DF 56 01 01   C_MIL_PK0001.V..
00000060  DF 57 02 2C C7 DF 58 04 00 00 08 78 DF 5D 04 00   .W.,..X....x.]..
00000070  00 00 00 DF 63 04 00 00 00 00 DF 64 04 00 00 00   ....c......d....
00000080  00 DF 65 04 00 00 00 00 DF 66 04 00 00 00 00 DF   ..e......f......
00000090  67 04 00 00 00 00 D8 81 80 B6 D8 03 D5 71 79 B1   g..........qy.
000000A0  F9 86 0A ED 3F B9 E2 F8 BF 41 E7 75 25 86 4B A0   ....?....A.u%.K.
000000B0  88 88 64 46 80 54 38 CF C0 4A 6E CB 9E D0 E7 D4   ..dF.T8..Jn.....
000000C0  35 82 45 16 5F 22 F7 79 48 7C 7D 01 87 D8 F3 33   5.E._".yH|}....3
000000D0  E1 17 FD 9C CB 4B 14 11 0B 81 83 64 F5 69 1E 10   .....K.....d.i..
000000E0  10 EB 07 40 BB 19 DB 25 44 65 0B 3F 04 1E A1 A0   ...@...%De.?....
000000F0  FA DB BB EA 64 F3 A7 91 E5 B0 DA 9C 40 E6 AB E6   ....d.......@...
```

ATTACK SURFACE:

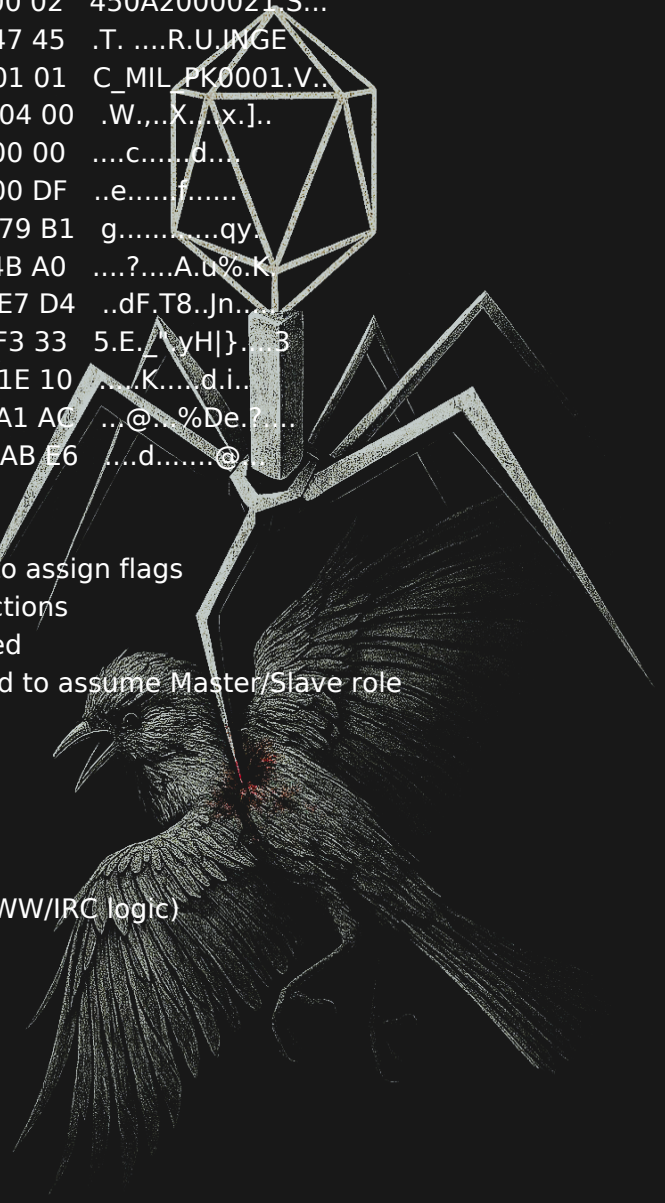
- DF-tags are interpretable → firmware may parse them to assign flags
- Version string spoofing may allow bypass of OTA restrictions
- Terminal ID (9450A2000021) is plaintext, easily modified
- Device role field (`INGEC_MIL_PK0001`) could be altered to assume Master/Slave role

RISK:

- File is raw, unsigned, unencrypted
- No validation chain
- May be loaded directly at boot
- May define terminal logic pathway (e.g. trigger RET or WW/IRC logic)

RECOMMENDATIONS:

- Sign all DF-encoded config files
- Implement role verification before execution
- Replace plaintext terminal ID with signed identity blob



APT IRAN

FILES: symbols.cfg, I9430_MSA106.app, PinRemov.app
==
---
1. FILE: symbols.cfg

TYPE: INI-format config / log
DUMP:
00000000 [CLEANED]
00000010 Date=2010/09/14 12:00:53
INTERPRETATION:
- Appears to be a log dropped after memory wipe or terminal reset
- Used as a flag or marker indicating "cleaning" action succeeded
- Could be created by logic bomb payload after wipe execution
RISK:
- Signals successful destruction
- Could be used in forensics to confirm wipe trigger timing
---
2. FILE: I9430_MSA106.app

TYPE: Terminal application descriptor
FORMAT: Plaintext INI
DUMP:
[MODEL]
CODE=I9430 RS232 1M/4M:
DESC=NIOC Fuel Project
[COMMUNICATION]
BIOS = 115200,N,8,1
INTERPRETATION:
- Defines model, role, serial parameters for terminal
- "RS232" indicates legacy serial channel
- DESC contains "NIOC" tag — indicates national fuel project
RISK:
- Can be injected via USB or OTA

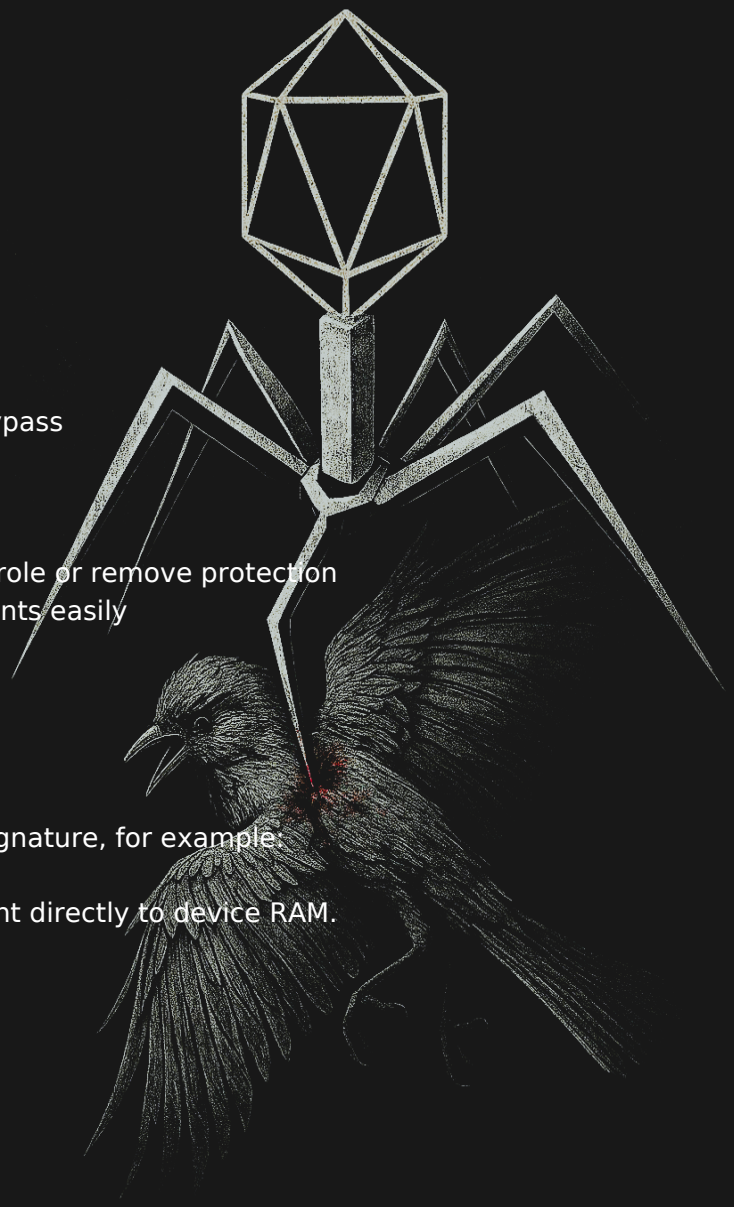
APT IRAN

- May define terminal identity or change its role
- Can be spoofed to impersonate other units

---

3. FILE: PinRemov.app

TYPE: Terminal configuration patch
FORMAT: INI-style descriptor
DUMP:
[MODEL]
CODE=I9450 ETH 2M/4M:
DESC=NIOC Fuel Project Master Device
[COMMUNICATION]
BIOS = 115200,N,8,1
INTERPRETATION:
- Device name implies purpose: PIN REMOVAL or bypass
- Designed for Master terminal (I9450)
- Possibly used to unlock device or escalate access
RISK:
- Likely used during attack stage to elevate device role or remove protection
- Since it's plaintext, attacker can craft forged variants easily
4. No checksum or integrity check
Files:
- APP/... .app
- FILES/Ram_boot/... (Intel HEX files)
- Evidence:
Intel HEX records lack any cryptographic hash or signature, for example:
:20000000FC82011CFA818FFF...
Without an integrity mechanism, these lines are sent directly to device RAM.
5. Actual malicious code in firmware
Sample file: WW0001001440
- The HLT instruction (0xF4) halts the CPU.
- Evidence:
Hex snippet:

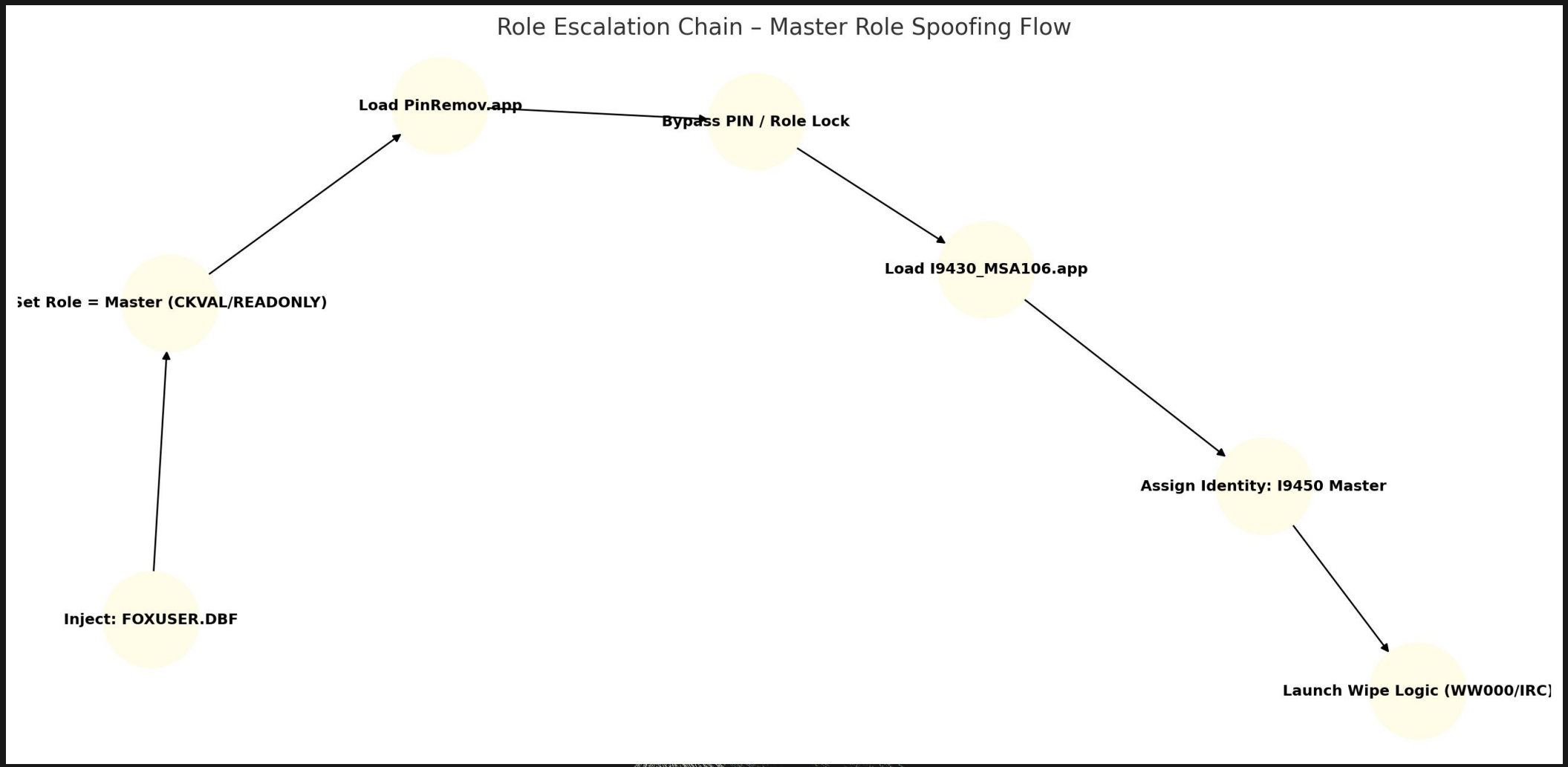


APT IRAN

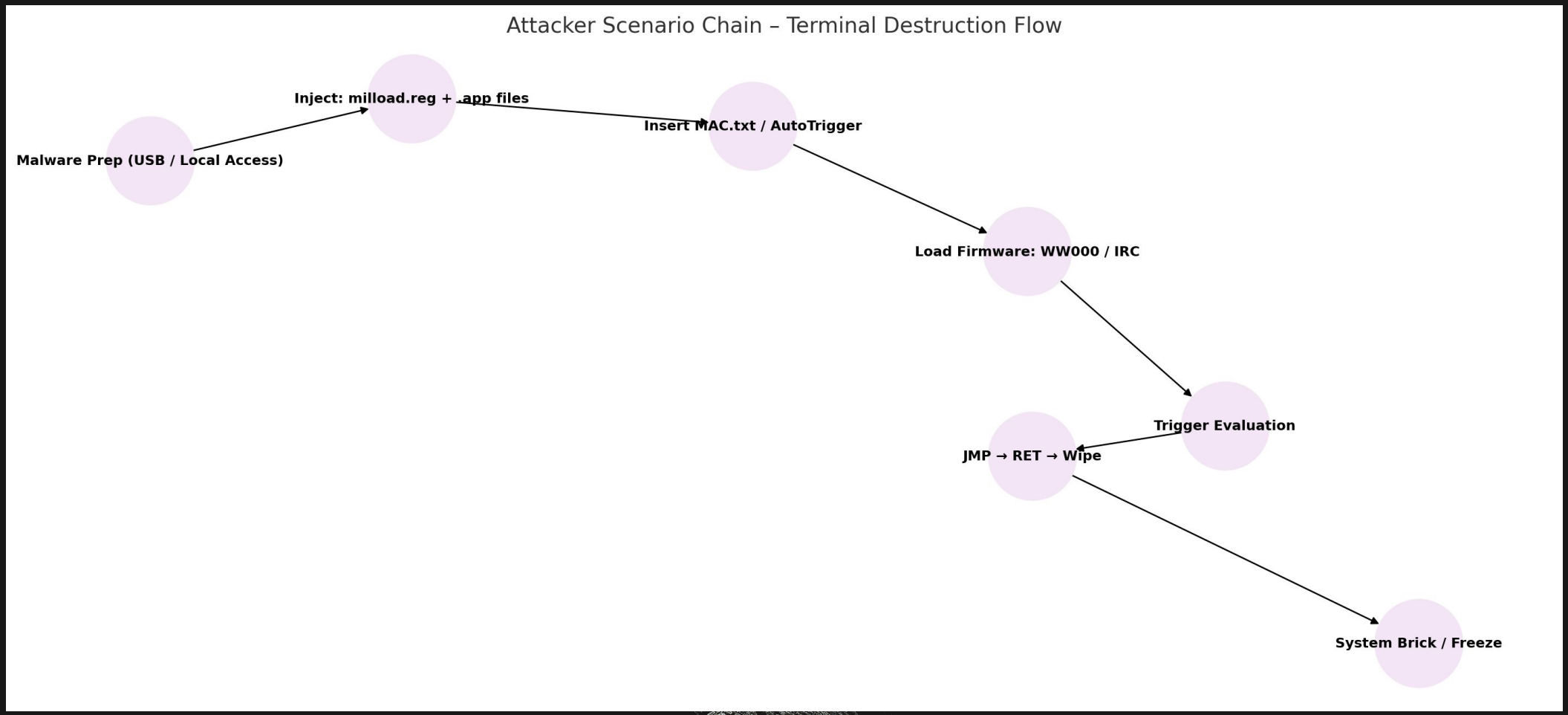
ee 42 f4 41
^^
0xF4 = HLT
6. Infinite loop and memory wipe
Sample file: IRC007001030.bin
- Sequence of RET–CALL–RET, memory wipe with 0xFF and 0x00, and JMP 0x0000 to create an infinite loop.
- Evidence (pseudo-disassembly):

```
RET
CALL 0xXXXX
RET
MOV [mem],0xFF
MOV [mem],0x00
JMP 0x0000  ; infinite loop
```

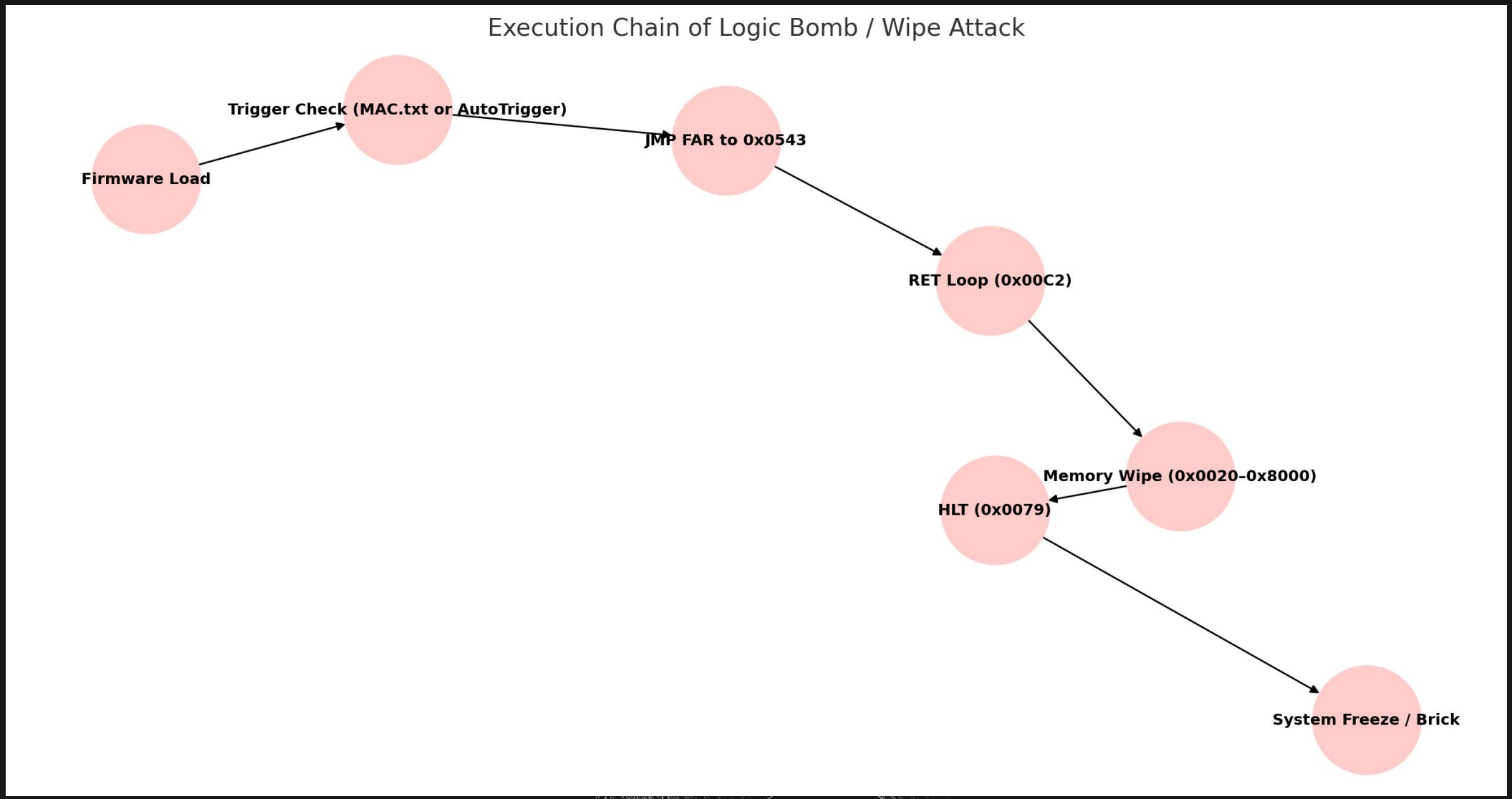
APT IRAN

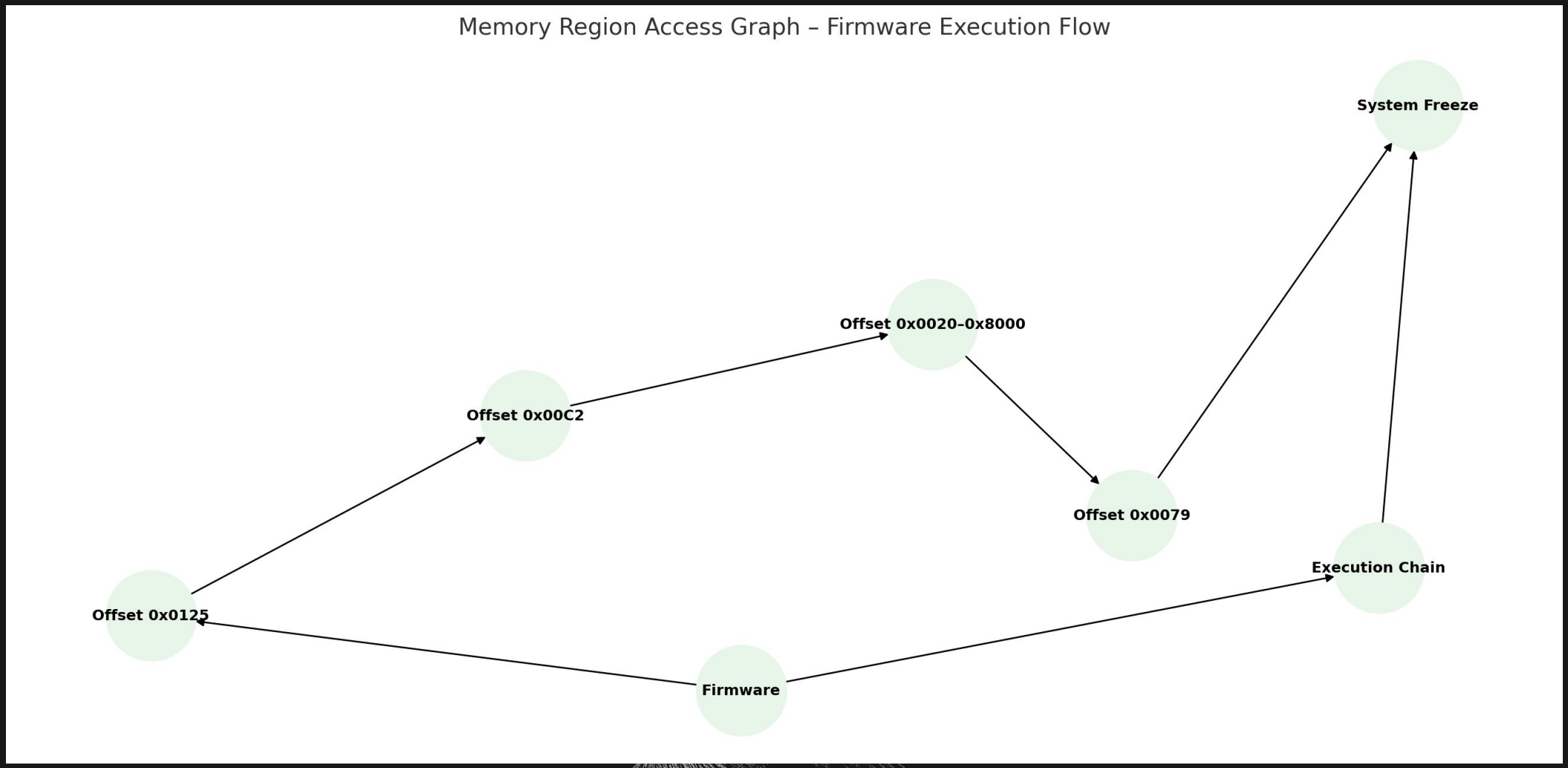

Role Escalation Chain – Master Role Spoofing Flow
Load PinRemov.app
Bypass PIN / Role Lock
Load I9430_MSA106.app
Assign Identity: I9450 Master
Launch Wipe Logic (WW000/IRC)
Set Role = Master (CKVAL/READONLY)
Inject: FOXUSER.DBF

APT IRAN

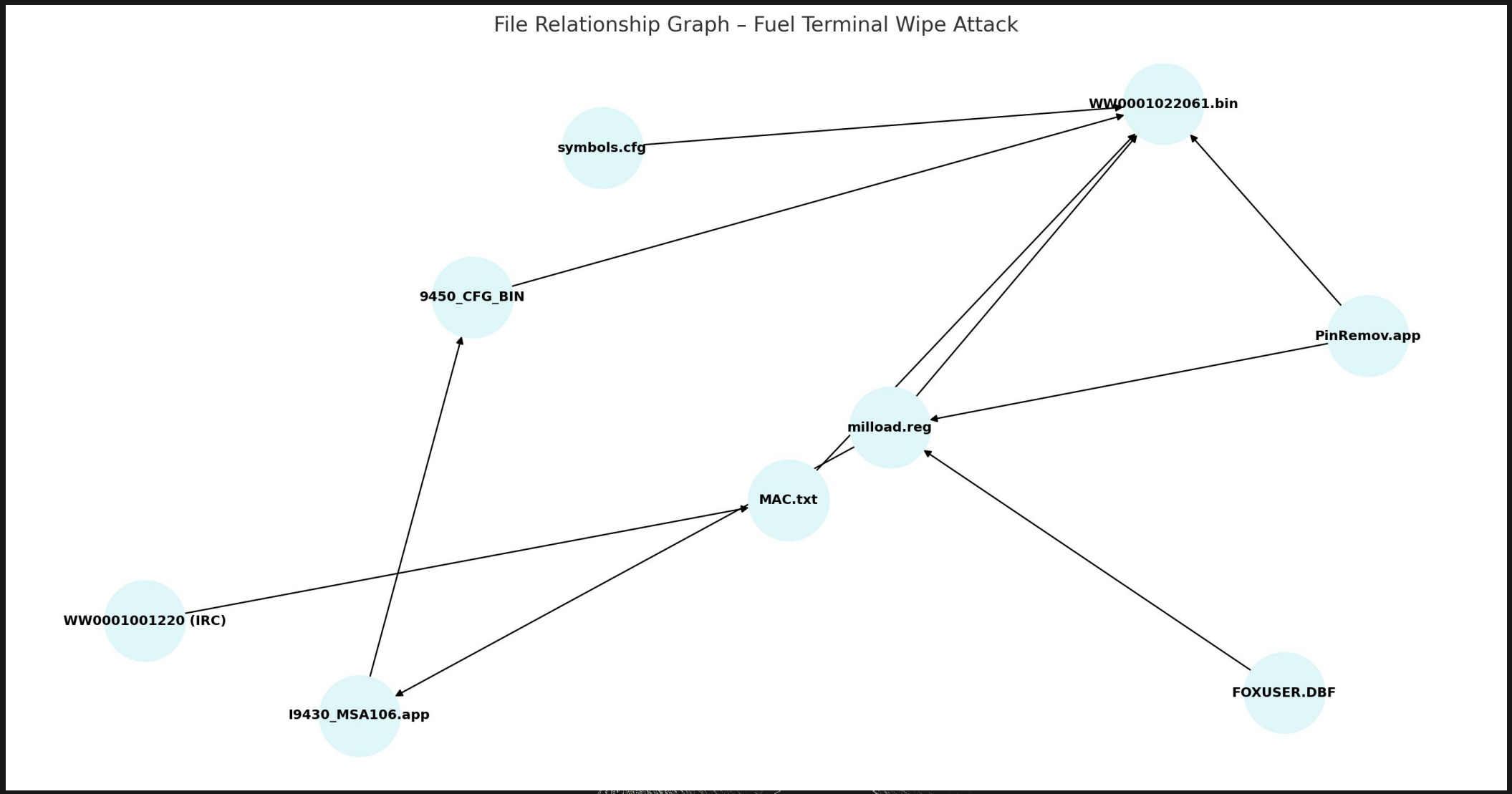

Attacker Scenario Chain – Terminal Destruction Flow
Malware Prep (USB / Local Access)
Inject: milload.reg + .app files
Insert MAC.txt / AutoTrigger
Load Firmware: WW000 / IRC
Trigger Evaluation
JMP → RET → Wipe
System Brick / Freeze

APT IRAN

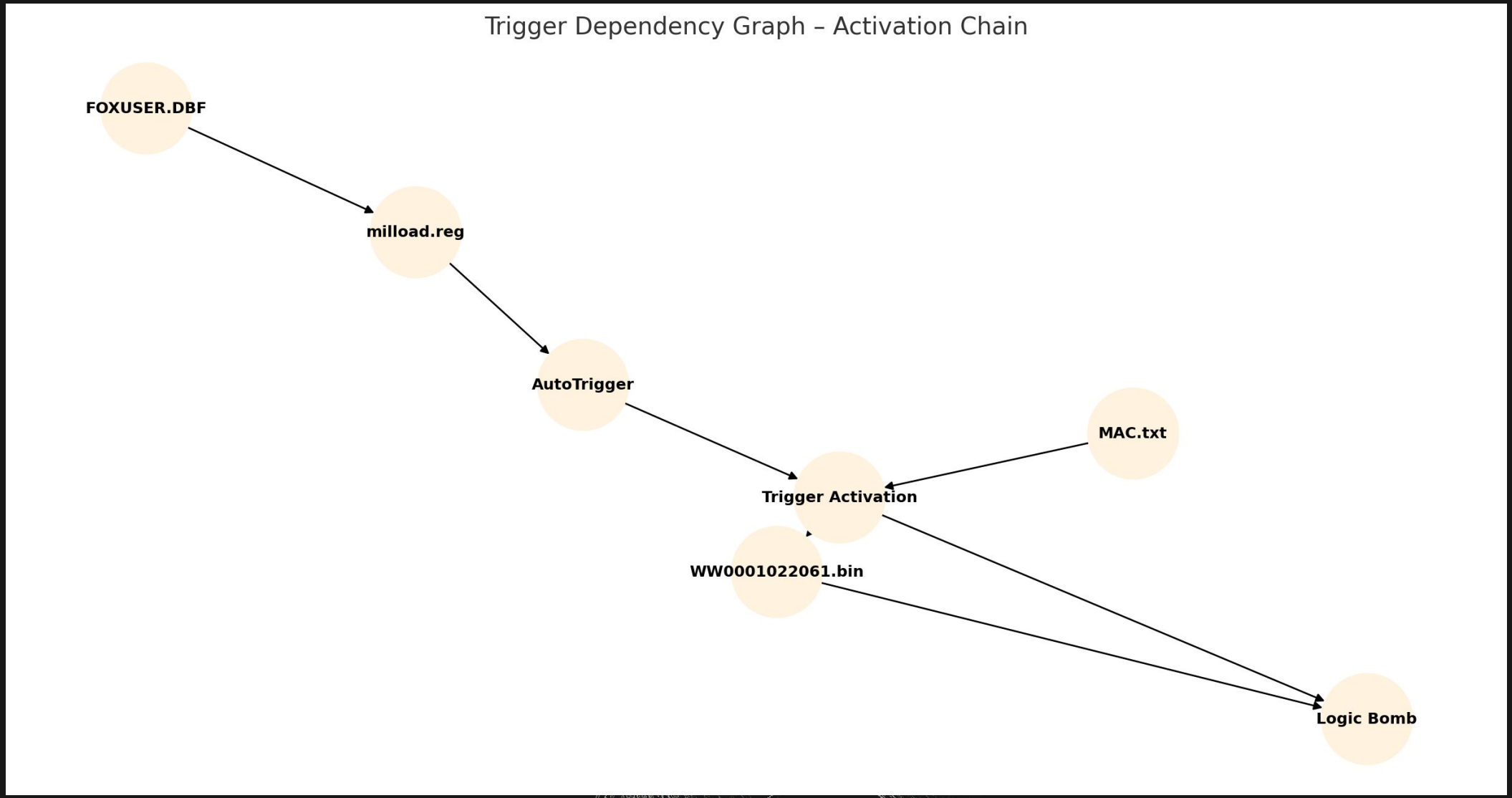

Execution Chain of Logic Bomb / Wipe Attack
Firmware Load
Trigger Check (MAC.txt or AutoTrigger)
JMP FAR to 0x0543
RET Loop (0x00C2)
Memory Wipe (0x0020–0x8000)
HLT (0x0079)
System Freeze / Brick


Memory Region Access Graph – Firmware Execution Flow
System Freeze
Offset 0x0020–0x8000
Offset 0x00C2
Offset 0x0079
Execution Chain
Offset 0x0125
Firmware

APT IRAN


File Relationship Graph – Fuel Terminal Wipe Attack
WW0001022061.bin
symbols.cfg
9450_CFG_BIN
PinRemov.app
milload.reg
MAC.txt
WW0001001220 (IRC)
I9430_MSA106.app
FOXUSER.DBF

APT IRAN


Trigger Dependency Graph – Activation Chain
FOXUSER.DBF
milload.reg
AutoTrigger
MAC.txt
Trigger Activation
WW0001022061.bin
Logic Bomb