Level 0x05

Offensive Security Researcher

Topics

- Kevin F
- Offensive Security Life

Kevin Finisterre

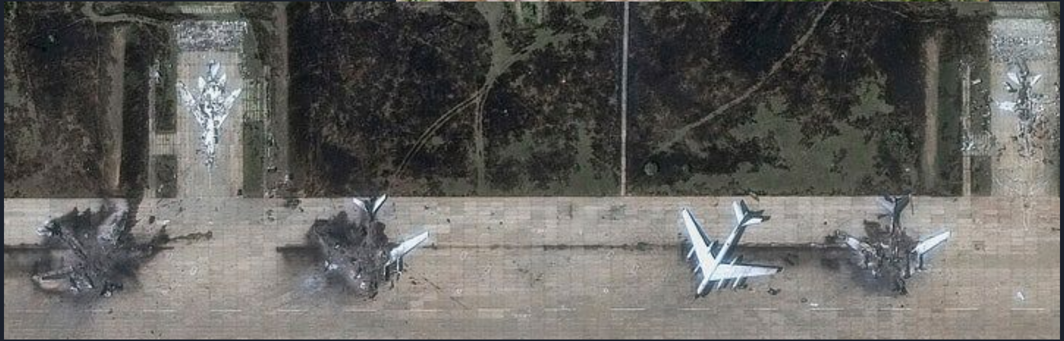
- Drone Hacking "OG"
  - Kickstarted drone hacking scene
    - I used a lot of his tools and guides when drone hacking myself
  - https://www.guinnpartners.com/kevin-finisterre-department-13/
  - Firmware decryption tools
  - Communication tools
  - Exposed many DJI abuses / privacy issues
    - Leaked photos
      - It's hard to find photos of Kevin, cause most of his photos are leaked DJI drone photos
    - User tracking

Life of a Offensive Cyber Engineer

- Offensive Security
  - Trying to hack / break into systems
  - For US government (else usually illegal)
- Targets
  - Adversary militaries or governments
  - Terrorists
  - Could we target US citizens?
- Objectives
  - Intelligence / Data collection
  - 5 Ds: Deny, Degrade, Disrupt, Deceive, or Destroy
  - Persist

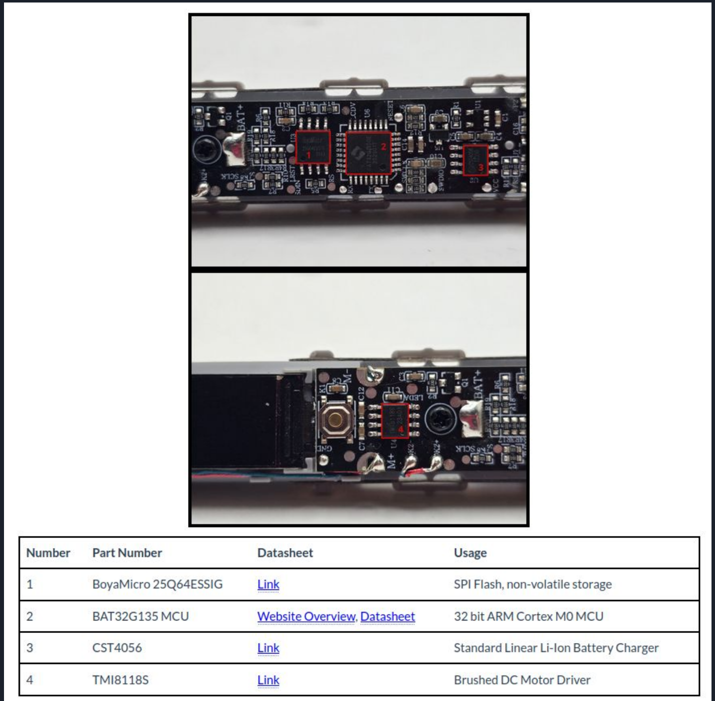
Example Case: Drones

- Air, sea, land, they are everywhere
- Commercial, agriculture, military / purpose built
- Hardware
  - Advanced computers
  - RF, Wi-Fi communication
  - Advanced Imagery / Camera
  - GPS / GLONASS positioning
  - Phones / display computers
- Intelligence?
- Threat?
  - Can kill personnel
  - Can destroy equipment
  - Can destroy infrastructure
  - Can surveil forces



Reverse Engineer Hardware

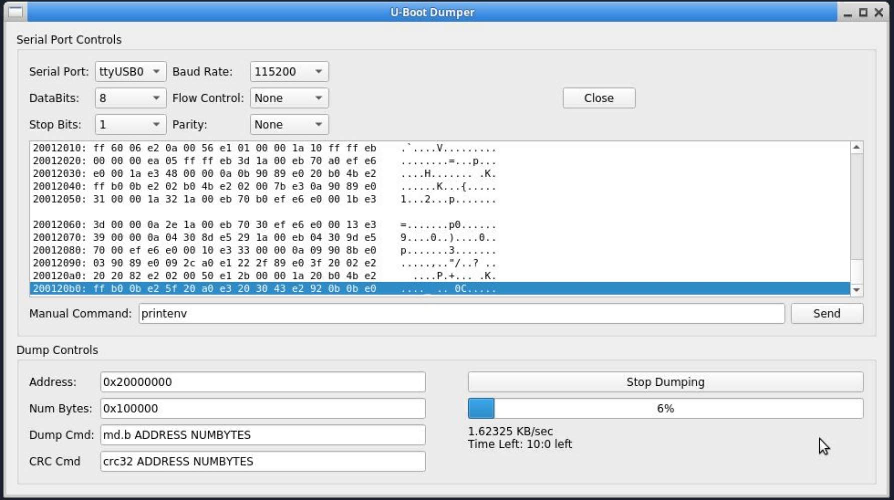
- Identification of hardware
  - Processors / capabilities
  - Memory devices
  - Communication interfaces
- Extract memory
  - So we can reverse engineer software
  - Hidden secrets, password hashes, keys
  - Logs
- Interfaces
  - Capture messages between devices
  - UART - Debug interfaces / console
  - JTAG - talk to CPU
  - RF capabilities
- Research



Number	Part Number	Datasheet	Usage
1	BoyaMicro 25Q64ESSIG	Link	SPI Flash, non-volatile storage
2	BAT32G135 MCU	Website Overview, Datasheet	32 bit ARM Cortex M0 MCU
3	CST4056	Link	Standard Linear Li-Ion Battery Charger
4	TMI8118S	Link	Brushed DC Motor Driver

Extracting and Capturing

- Desolder / In-circuit interrogation
- JTAG
- UART



U-Boot Dumper

Serial Port Controls

Serial Port: ttyUSB0 Baud Rate: 115200

DataBits: 8 Flow Control: None

Stop Bits: 1 Parity: None

```
20012010: ff 60 06 e2 0a 00 56 e1 01 00 00 1a 10 ff ff eb    .`....V.........
20012020: 00 00 00 ea 05 ff ff eb 3d 1a 00 eb 70 a0 ef e6    ........=...p...
20012030: e0 00 1a e3 48 00 00 0a 0b 90 89 e0 20 b0 4b e2    ....H....... .K.
20012040: ff b0 0b e2 02 b0 4b e2 02 00 7b e3 0a 90 89 e0    ......K...{.....
20012050: 31 00 00 1a 32 1a 00 eb 70 b0 ef e6 e0 00 1b e3    1...2...p.......

20012060: 3d 00 00 0a 2e 1a 00 eb 70 30 ef e6 e0 00 13 e3    =.......p0......
20012070: 39 00 00 0a 04 30 8d e5 29 1a 00 eb 04 30 9d e5    9....0..)....0..
20012080: 70 00 ef e6 e0 00 10 e3 33 00 00 0a 09 90 8b e0    p.......3.......
20012090: 03 90 89 e0 09 2c a0 e1 22 2f 89 e0 3f 20 02 e2    .....,.."/..? ..
200120a0: 20 20 82 e2 02 00 50 e1 2b 00 00 1a 20 b0 4b e2      ....P.+... .K.
200120b0: ff b0 0b e2 5f 20 a0 e3 20 30 43 e2 92 0b 0b e0    ....  .. 0C.....
```

Manual Command: printenv

Dump Controls

Address: 0x20000000

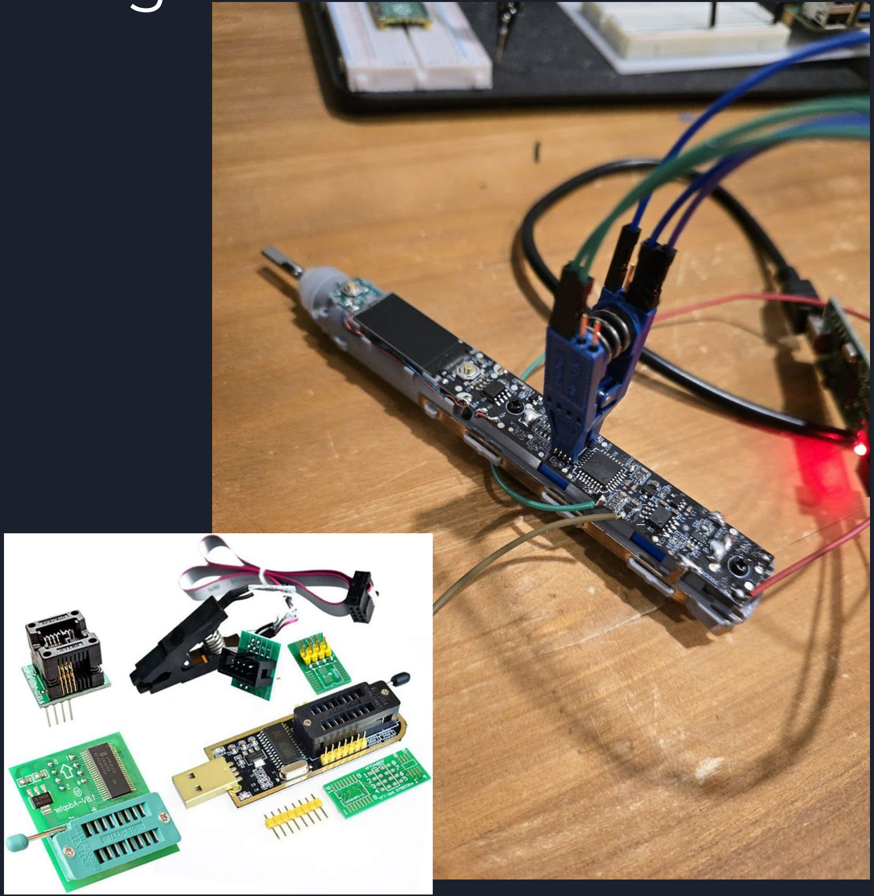
Num Bytes: 0x100000

Dump Cmd: md.b ADDRESS NUMBYTES

CRC Cmd crc32 ADDRESS NUMBYTES

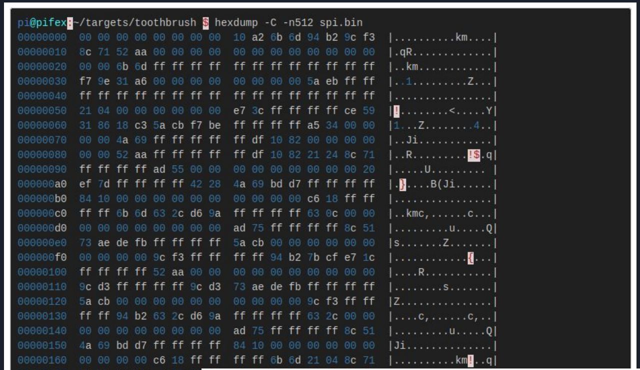
6%

1.62325 KB/sec
Time Left: 10:0 left

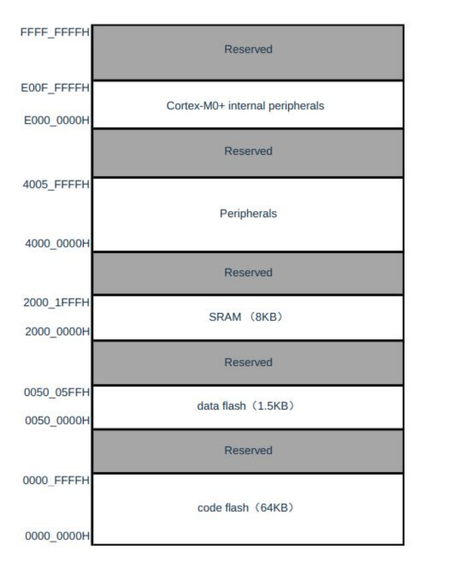


Analyze Memory Contents

- Code
  - Boot code
  - Application code
    - What memory address?
- Data
  - Encrypted?
  - File system?
    - Can I extract files?
    - Logs?
    - Password file?
    - Permissions?

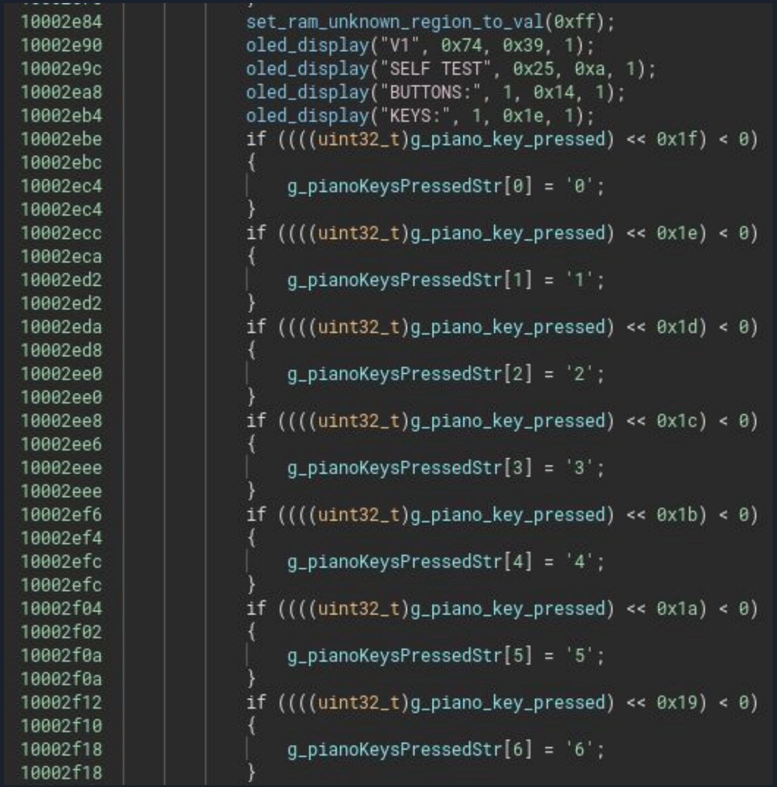


```
pi@pifex:~/targets/toothbrush $ hexdump -C -n512 spi.bin
00000000  00 00 00 00 00 00 00 00  10 a2 6b 6d 94 b2 9c f3  |..........km....|
00000010  8c 71 52 aa 00 00 00 00  00 00 00 00 00 00 00 00  |.qR.............|
00000020  00 00 6b 6d ff ff ff ff  ff ff ff ff ff ff ff ff  |..km............|
00000030  f7 9e 31 a6 00 00 00 00  00 00 00 00 5a eb ff ff  |..1.........Z...|
00000040  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
00000050  21 04 00 00 00 00 00 00  e7 3c ff ff ff ff ce 59  |!........<.....Y|
00000060  31 86 18 c3 5a cb f7 be  ff ff ff ff a5 34 00 00  |1...Z........4..|
00000070  00 00 4a 69 ff ff ff ff  ff df 10 82 00 00 00 00  |..Ji............|
00000080  00 00 52 aa ff ff ff ff  ff df 10 82 21 24 8c 71  |..R.........!$.q|
00000090  ff ff ff ff ad 55 00 00  00 00 00 00 00 00 00 20  |.....U.......... |
000000a0  ef 7d ff ff ff ff 42 28  4a 69 bd d7 ff ff ff ff  |.}....B(Ji......|
000000b0  84 10 00 00 00 00 00 00  00 00 00 00 c6 18 ff ff  |................|
000000c0  ff ff 6b 6d 63 2c d6 9a  ff ff ff ff 63 0c 00 00  |..kmc,......c...|
000000d0  00 00 00 00 00 00 00 00  ad 75 ff ff ff ff 8c 51  |.........u.....Q|
000000e0  73 ae de fb ff ff ff ff  5a cb 00 00 00 00 00 00  |s.......Z.......|
000000f0  00 00 00 00 9c f3 ff ff  ff ff 94 b2 7b cf e7 1c  |............{...|
00000100  ff ff ff ff 52 aa 00 00  00 00 00 00 00 00 00 00  |....R...........|
00000110  9c d3 ff ff ff ff 9c d3  73 ae de fb ff ff ff ff  |........s.......|
00000120  5a cb 00 00 00 00 00 00  00 00 00 00 9c f3 ff ff  |Z...............|
00000130  ff ff 94 b2 63 2c d6 9a  ff ff ff ff 63 2c 00 00  |....c,......c,..|
00000140  00 00 00 00 00 00 00 00  ad 75 ff ff ff ff 8c 51  |.........u.....Q|
00000150  4a 69 bd d7 ff ff ff ff  84 10 00 00 00 00 00 00  |Ji..............|
00000160  00 00 00 00 c6 18 ff ff  ff ff 6b 6d 21 04 8c 71  |..........km!..q|
00000170  ff ff ff ff ad 75 00
00000180  f7 9e ff ff ff ff 42
00000190  ff df 10 a2 00 00 00
000001a0  ff df 08 61 00 00 00
000001b0  39 e7 21 04 63 0c f7
000001c0  00 00 00 00 5a cb ff
000001d0  ff ff ff ff ff ff ff
000001e0  00 00 63 2c ff ff ff
000001f0  ef 7d 29 65 00 00 00
```



Static Reverse Engineering

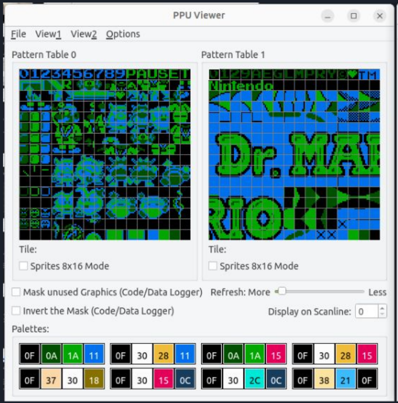
- Strings
  - Program output
  - Commands / messages
- Disassemble
  - Binary Ninja / Ghidra / IDA
  - Decompiler
- Understand code
  - Function names
  - Data structures
  - Software flow / modes
- Identify Encryption
- Hidden features?
  - Debug modes?
  - Backdoors?



```
10002e84        set_ram_unknown_region_to_val(0xff);
10002e90        oled_display("V1", 0x74, 0x39, 1);
10002e9c        oled_display("SELF TEST", 0x25, 0xa, 1);
10002ea8        oled_display("BUTTONS:", 1, 0x14, 1);
10002eb4        oled_display("KEYS:", 1, 0x1e, 1);
10002ebe        if ((((uint32_t)g_piano_key_pressed) << 0x1f) < 0)
10002ebc        {
10002ec4            g_pianoKeysPressedStr[0] = '0';
10002ec4        }
10002ecc        if ((((uint32_t)g_piano_key_pressed) << 0x1e) < 0)
10002eca        {
10002ed2            g_pianoKeysPressedStr[1] = '1';
10002ed2        }
10002eda        if ((((uint32_t)g_piano_key_pressed) << 0x1d) < 0)
10002ed8        {
10002ee0            g_pianoKeysPressedStr[2] = '2';
10002ee0        }
10002ee8        if ((((uint32_t)g_piano_key_pressed) << 0x1c) < 0)
10002ee6        {
10002eee            g_pianoKeysPressedStr[3] = '3';
10002eee        }
10002ef6        if ((((uint32_t)g_piano_key_pressed) << 0x1b) < 0)
10002ef4        {
10002efc            g_pianoKeysPressedStr[4] = '4';
10002efc        }
10002f04        if ((((uint32_t)g_piano_key_pressed) << 0x1a) < 0)
10002f02        {
10002f0a            g_pianoKeysPressedStr[5] = '5';
10002f0a        }
10002f12        if ((((uint32_t)g_piano_key_pressed) << 0x19) < 0)
10002f10        {
10002f18            g_pianoKeysPressedStr[6] = '6';
10002f18        }
```

Dynamic Reverse Engineering

- Debug / see code as it runs
  - Can sometimes debug on target hardware
- Emulation
  - Write software to simulate how hardware runs software
  - Write tools to analyze software as it runs
    - Tile viewers, OS introspection



Pattern Table 0

Tile:

□ Sprites 8x16 Mode

Pattern Table 1

Tile:

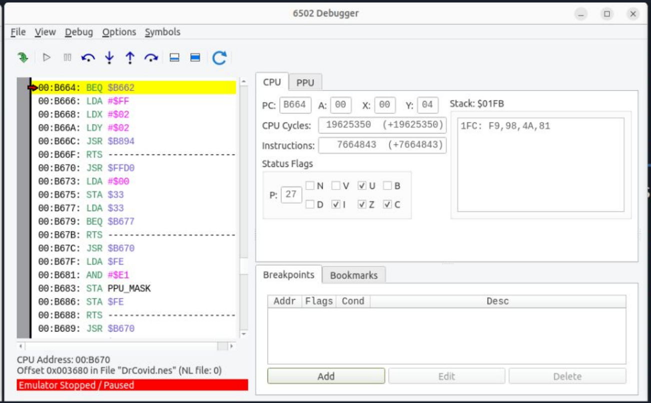
□ Sprites 8x16 Mode

□ Mask unused Graphics (Code/Data Logger)　Refresh: More　Less

□ Invert the Mask (Code/Data Logger)　Display on Scanline: 0

Palettes:

```
0F 0A 1A 11   0F 30 28 11   0F 0A 1A 15   0F 30 28 15
0F 37 30 18   0F 30 15 0C   0F 30 2C 0C   0F 38 21 0F
```



```
00:B664: BEQ $B662
00:B666: LDA #$FF
00:B668: LDX #$02
00:B66A: LDY #$02
00:B66C: JSR $B894
00:B66F: RTS -----------------------
00:B670: JSR $FFD0
00:B673: LDA #$00
00:B675: STA $33
00:B677: LDA $33
00:B679: BEQ $B677
00:B67B: RTS -----------------------
00:B67C: JSR $B670
00:B67F: LDA $FE
00:B681: AND #$E1
00:B683: STA PPU_MASK
00:B686: STA $FE
00:B688: RTS -----------------------
00:B689: JSR $B670
```

PC: B664　A: 00　X: 00　Y: 04　Stack: $01FB

CPU Cycles: 19625350 (+19625350)

Instructions: 7664843 (+7664843)

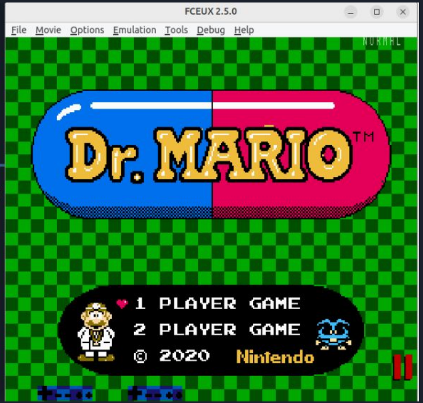
1FC: F9,98,4A,81

Status Flags

P: 27　□ N □ V ☑ U □ B
□ D ☑ I ☑ Z ☑ C

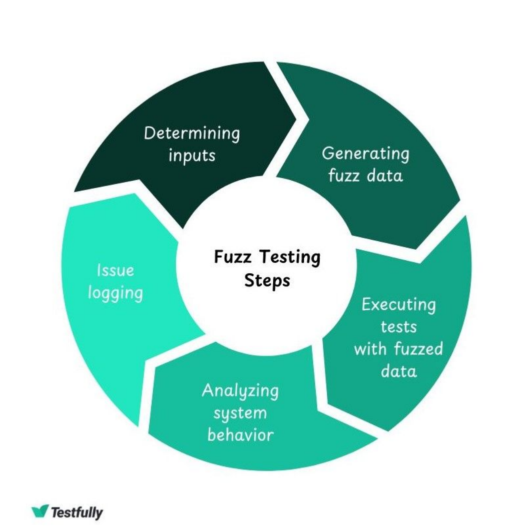
Addr	Flags	Cond	Desc

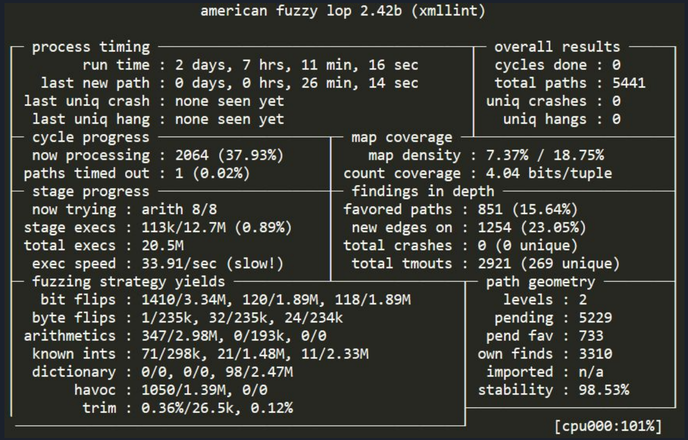
CPU Address: 00:B670
Offset 0x003680 in File "DrCovid.nes" (NL file: 0)
Emulator Stopped / Paused



Bug finding / Fuzzing

- Static research / using your brain
- Research existing bug databases
- Fuzzing - feed program corrupt data and cause it to crash
    - Corpus: files / messages to corrupt
    - Corruption
        - Random data
        - AI driven corruption
    - Crash Triage
        - Capture crash data (inputs and outputs)
        - Repeatable?
        - What went wrong
    - Where do we do this?
        - Target hardware
        - Emulators



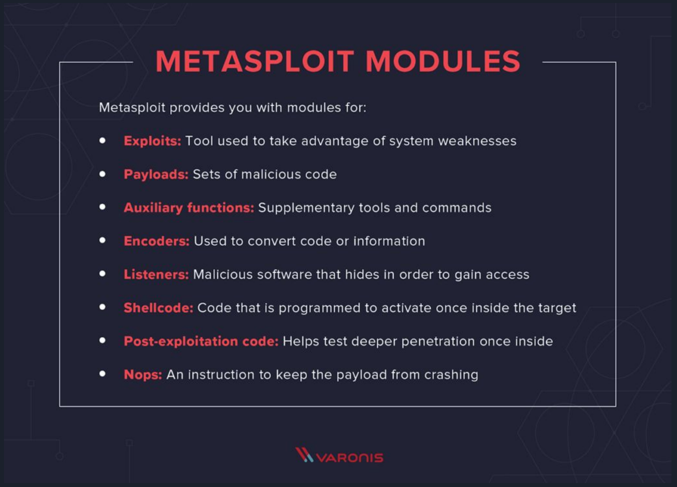


```
                  american fuzzy lop 2.42b (xmllint)

┌─ process timing ─────────────────────────────────────┬─ overall results ─┐
│        run time : 2 days, 7 hrs, 11 min, 16 sec      │  cycles done : 0  │
│   last new path : 0 days, 0 hrs, 26 min, 14 sec      │  total paths : 5441│
│ last uniq crash : none seen yet                      │ uniq crashes : 0  │
│  last uniq hang : none seen yet                      │   uniq hangs : 0  │
├─ cycle progress ────────────────────┬─ map coverage ─┴───────────────────┤
│  now processing : 2064 (37.93%)     │    map density : 7.37% / 18.75%    │
│ paths timed out : 1 (0.02%)         │ count coverage : 4.04 bits/tuple   │
├─ stage progress ────────────────────┼─ findings in depth ────────────────┤
│  now trying : arith 8/8             │ favored paths : 851 (15.64%)       │
│ stage execs : 113k/12.7M (0.89%)    │  new edges on : 1254 (23.05%)      │
│ total execs : 20.5M                 │ total crashes : 0 (0 unique)       │
│  exec speed : 33.91/sec (slow!)     │  total tmouts : 2921 (269 unique)  │
├─ fuzzing strategy yields ───────────┴───────────────┬─ path geometry ────┤
│   bit flips : 1410/3.34M, 120/1.89M, 118/1.89M      │    levels : 2      │
│  byte flips : 1/235k, 32/235k, 24/234k              │   pending : 5229   │
│ arithmetics : 347/2.98M, 0/193k, 0/0                │  pend fav : 733    │
│  known ints : 71/298k, 21/1.48M, 11/2.33M           │ own finds : 3310   │
│  dictionary : 0/0, 0/0, 98/2.47M                    │  imported : n/a    │
│       havoc : 1050/1.39M, 0/0                       │ stability : 98.53% │
│        trim : 0.36%/26.5k, 0.12%                    ├────────────────────┘
└─────────────────────────────────────────────────────┘          [cpu000:101%]
```

Exploitation and Development

- Implant: Convert your exploit into our code running on target
- Persistence: can we stay running on hardware after reset
- Communication: command and control of implant
- Effects:
  - Move onto network
  - Collect data
  - 5 D's
- Packaging: how does a soldier or operator use?
  - Hardware / Smartphone app?
  - USB drive / Rubber ducky
  - Transmit via wireless

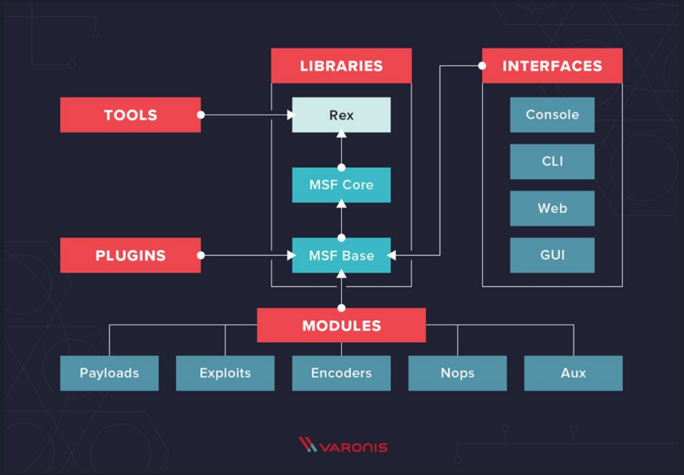


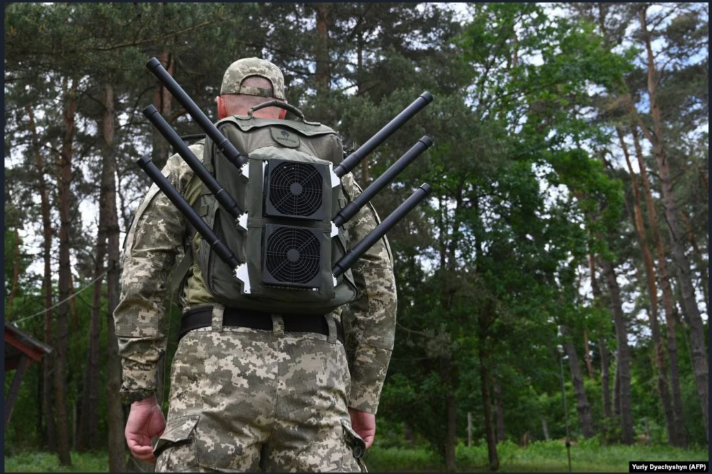
METASPLOIT MODULES

Metasploit provides you with modules for:

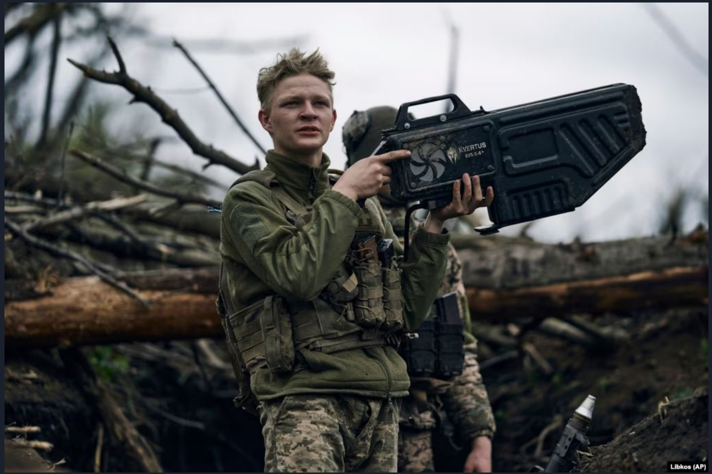
- **Exploits:** Tool used to take advantage of system weaknesses
- **Payloads:** Sets of malicious code
- **Auxiliary functions:** Supplementary tools and commands
- **Encoders:** Used to convert code or information
- **Listeners:** Malicious software that hides in order to gain access
- **Shellcode:** Code that is programmed to activate once inside the target
- **Post-exploitation code:** Helps test deeper penetration once inside
- **Nops:** An instruction to keep the payload from crashing

VARONIS

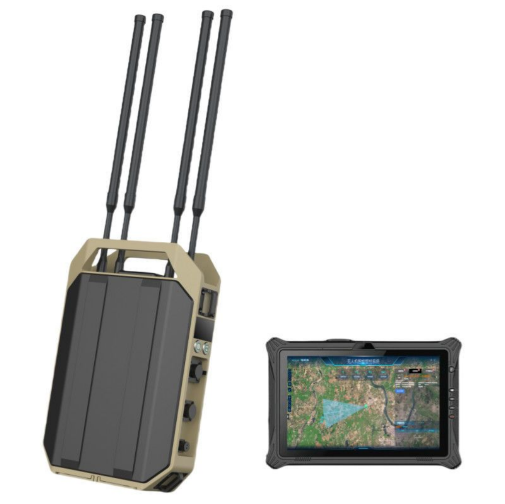




Yuriy Dyachyshyn (AFP)



Libkos (AP)



Links

- https://www.guinnpartners.com/kevin-finisterre-department-13/
- https://www.wired.com/beyond-the-beyond/2019/04/deny-degrade-disrupt-deceive-destroy/
- https://www.abc.net.au/news/2023-02-04/diy-weapons-innovation-drones-in-ukraine-war-russia/101910506
- https://www.foxnews.com/world/russia-vows-repair-planes-damaged-ukraine-massive-drone-attack-claims-were-not-destroyed
- https://voidstarsec.com/blog/brushing-up-part-2
- https://testfully.io/blog/fuzz-testing/
- https://www.rferl.org/a/anti-drone-evolution-ukraine-war-russia/33020303.html
- https://www.csmonitor.com/World/Middle-East/2011/1215/Exclusive-Iran-hijacked-US-drone-says-Iranian-engineer