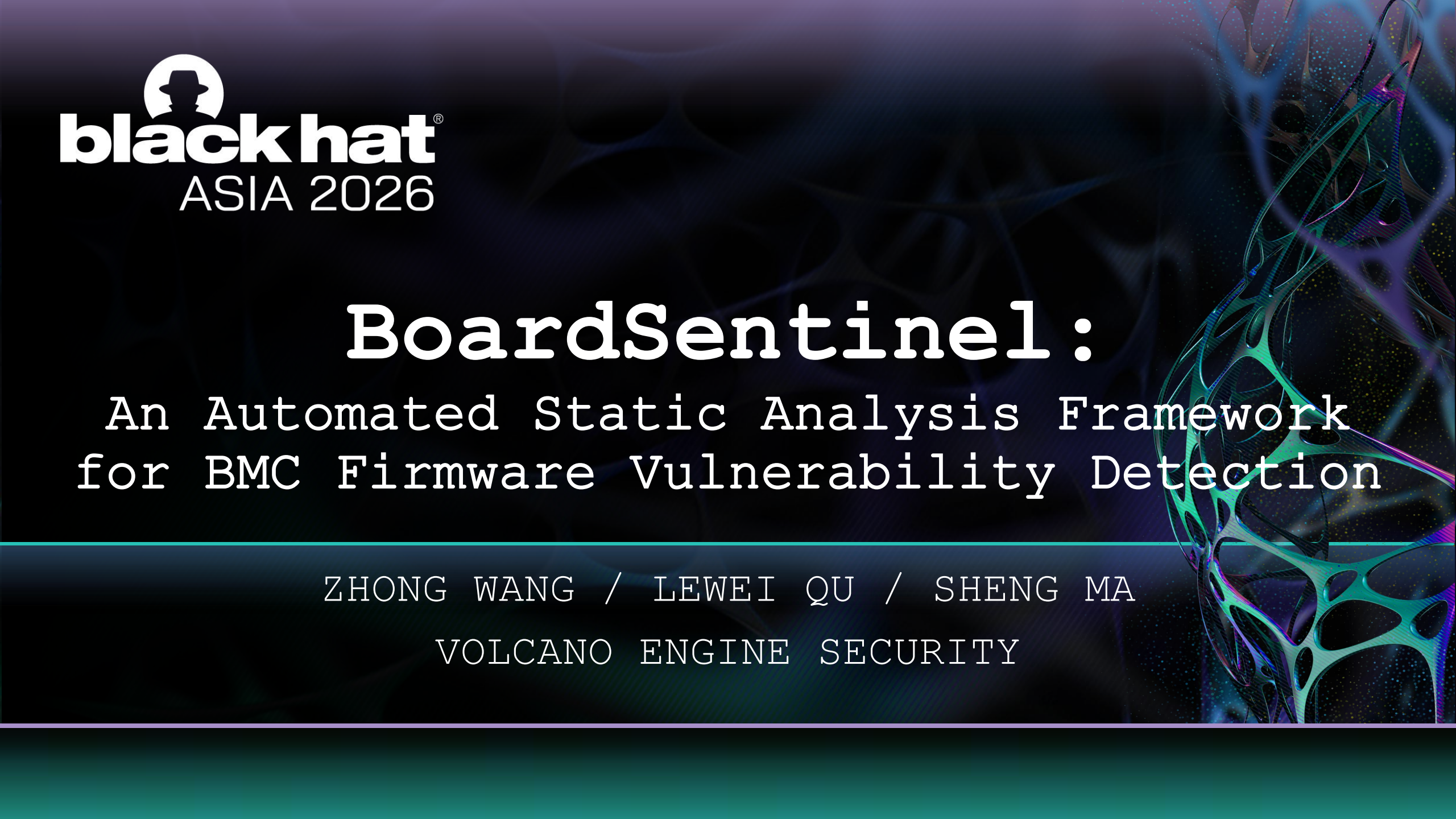


BoardSentinel:

An Automated Static Analysis Framework for BMC Firmware Vulnerability Detection

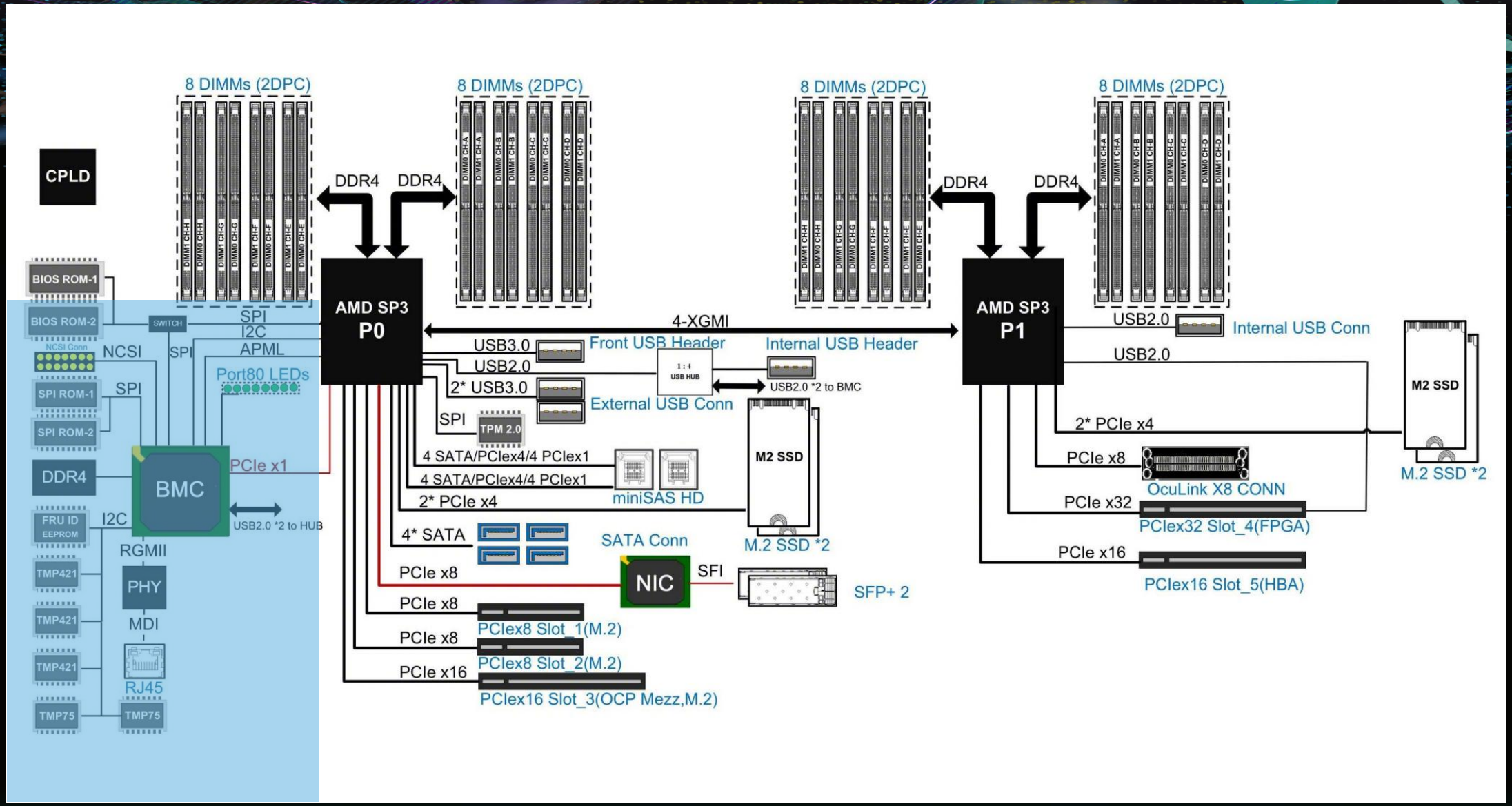
ZHONG WANG / LEWEI QU / SHENG MA

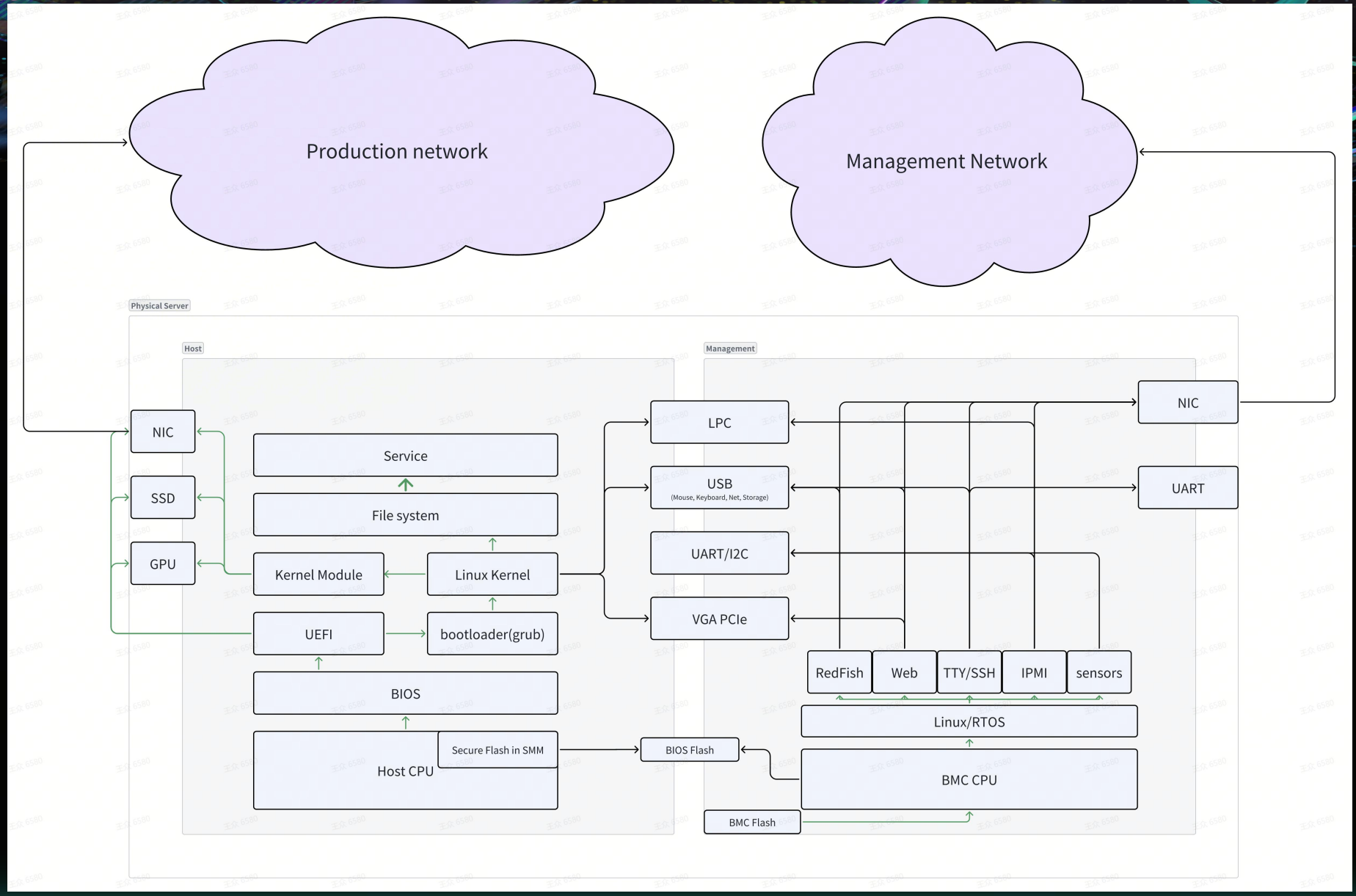
VOLCANO ENGINE SECURITY

$ whoami

- Speaker: Zhong Wang(@C0ss4ck)
  - Security Architect @ VolcEngine, specializing in IaaS and DaaS.
  - Security Researcher, focusing on firmware, OS, and containers.
  - Scavenger, collecting servers from x99 to genoa.
  - Github: C0ss4ck, X: CossackWang
- Team: Volcano Engine Security
  - responsible for
    - securing all Volcano Engine / BytePlus ToB businesses and cloud platform infrastructure covering security architecture, SDLC, vulnerability operations, security incident response, and security compliance
    - ensuring the safety of the Volcano Engine / BytePlus platform and empowering cloud business success.

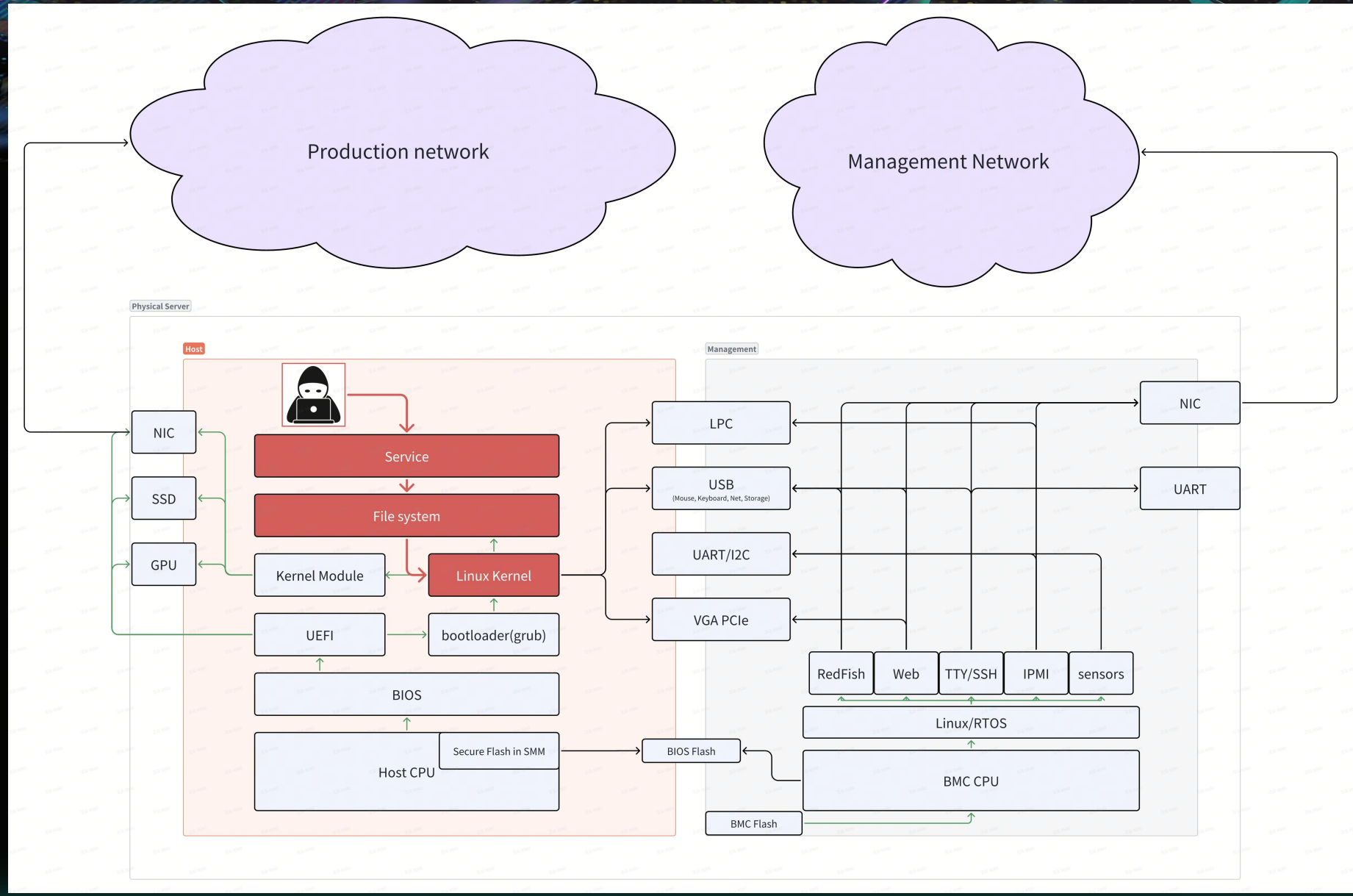
WHAT IS BMC?

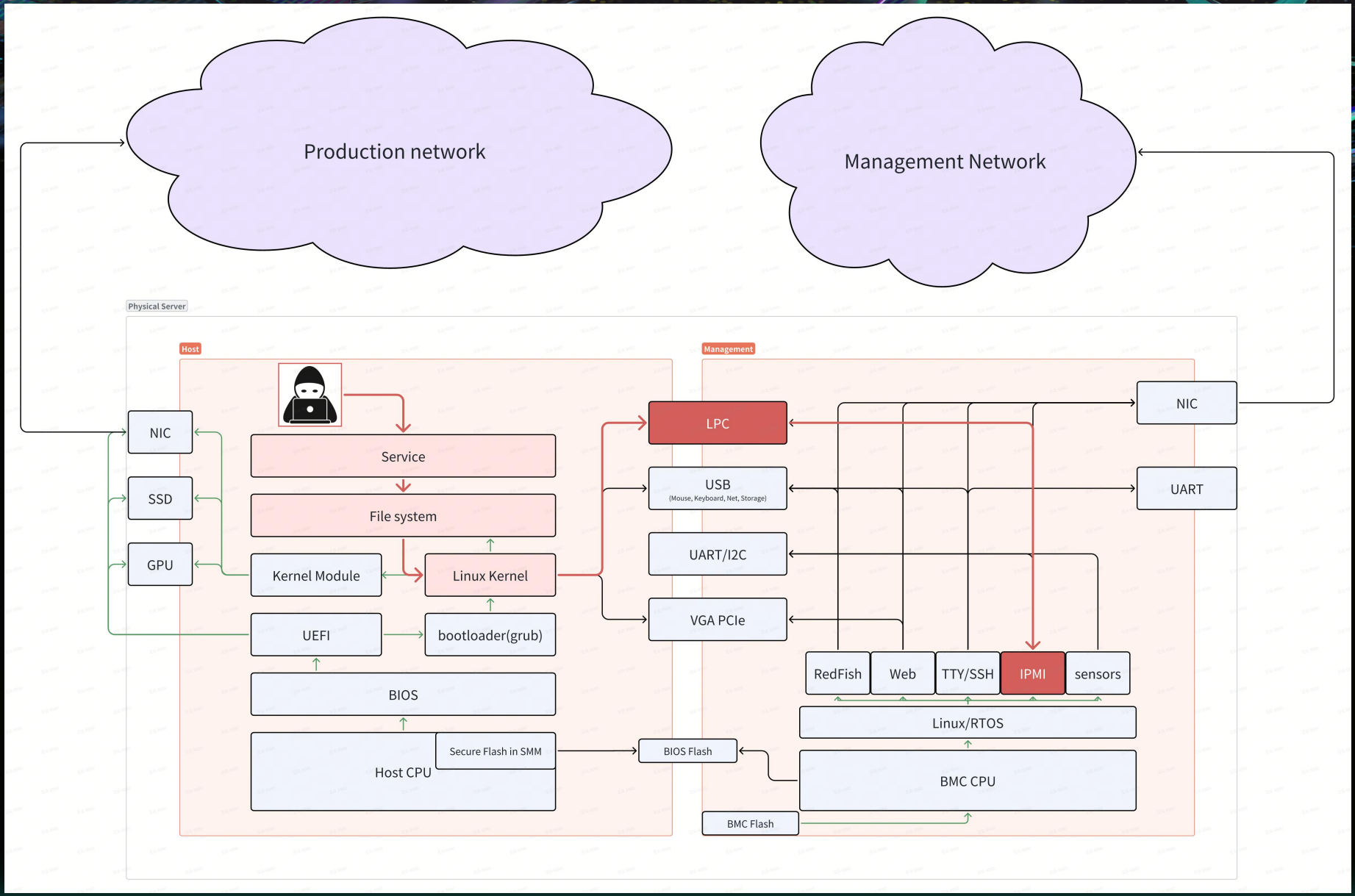


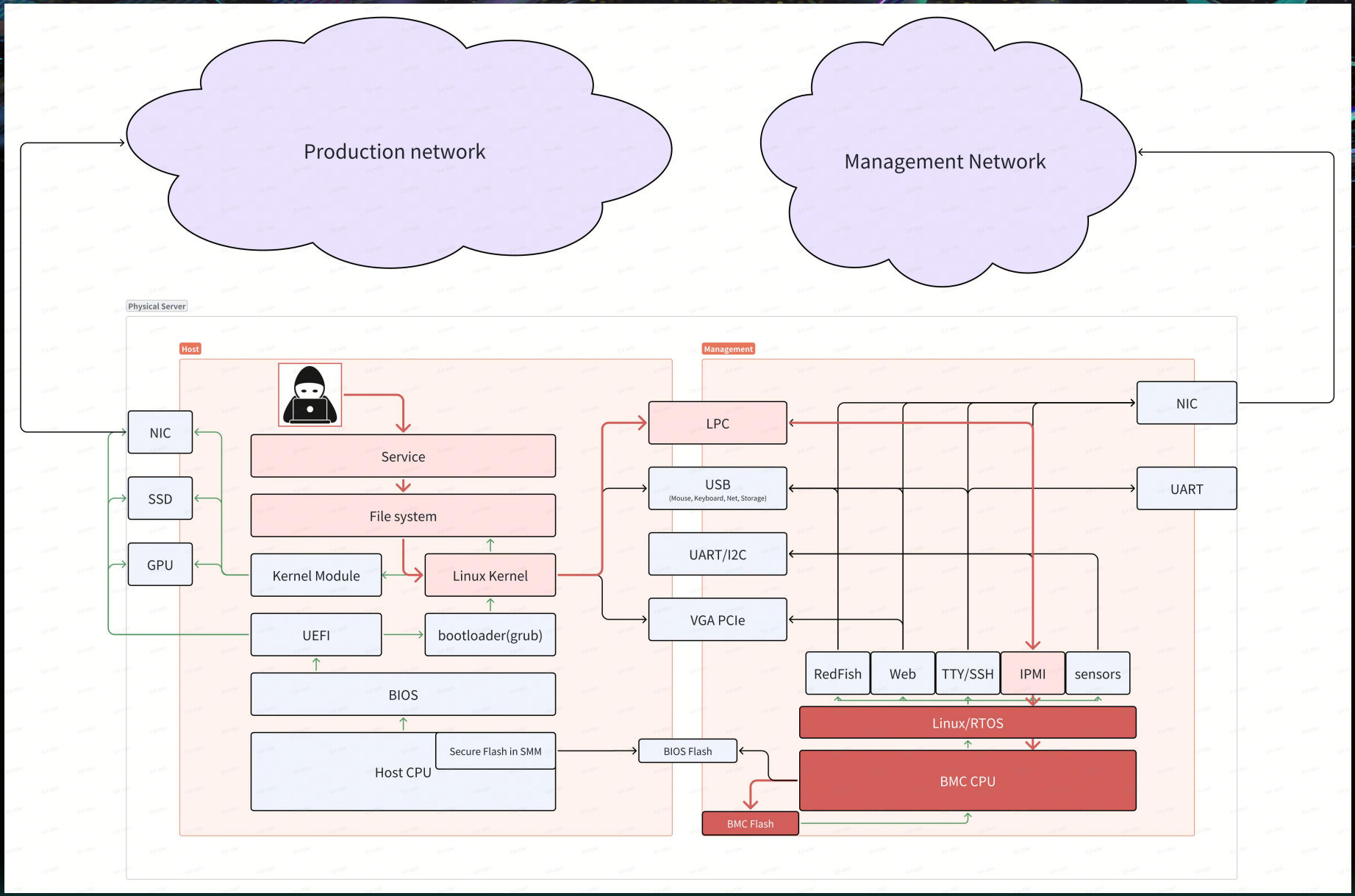


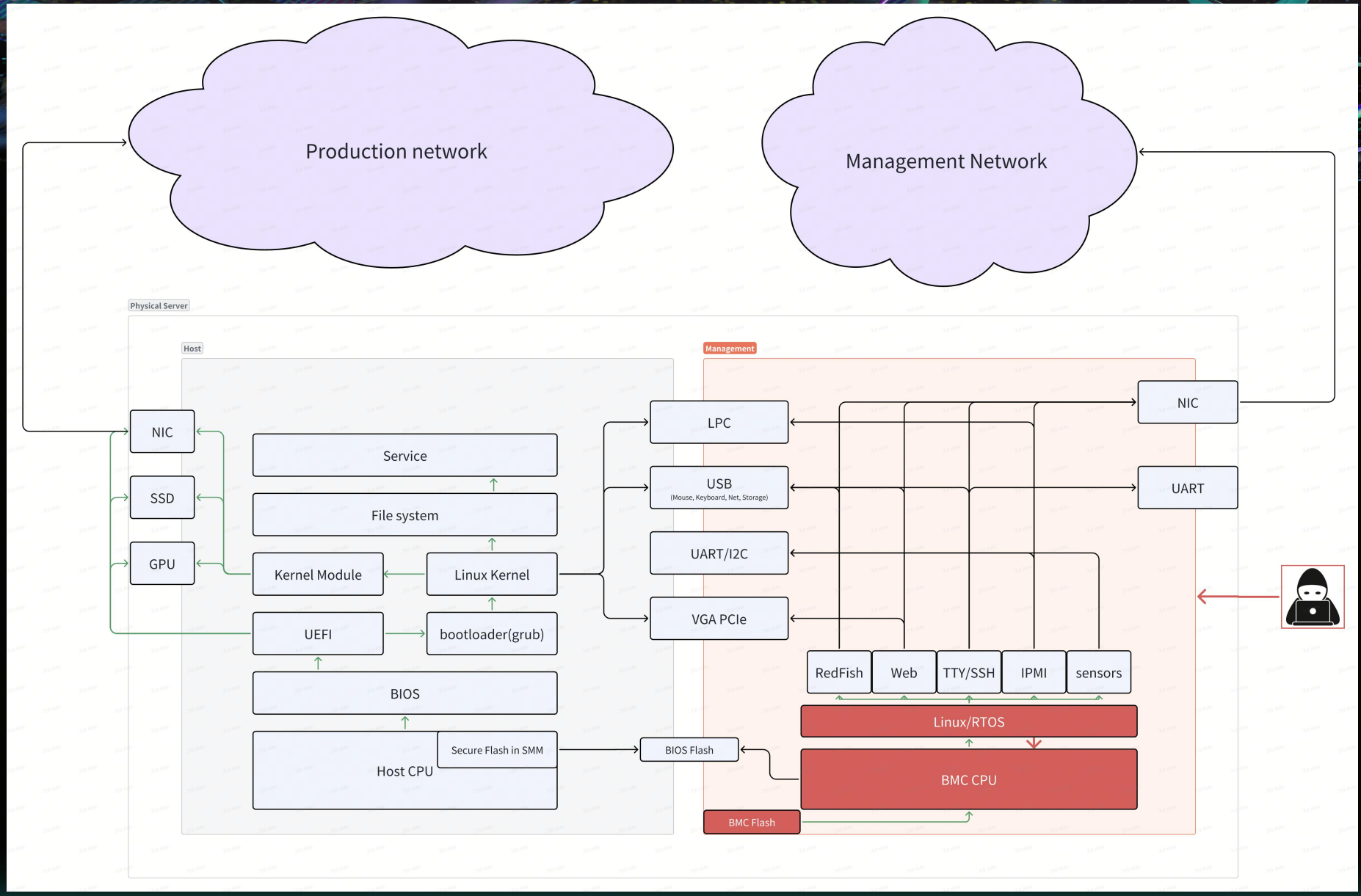
HOW TO BREAK BMC?

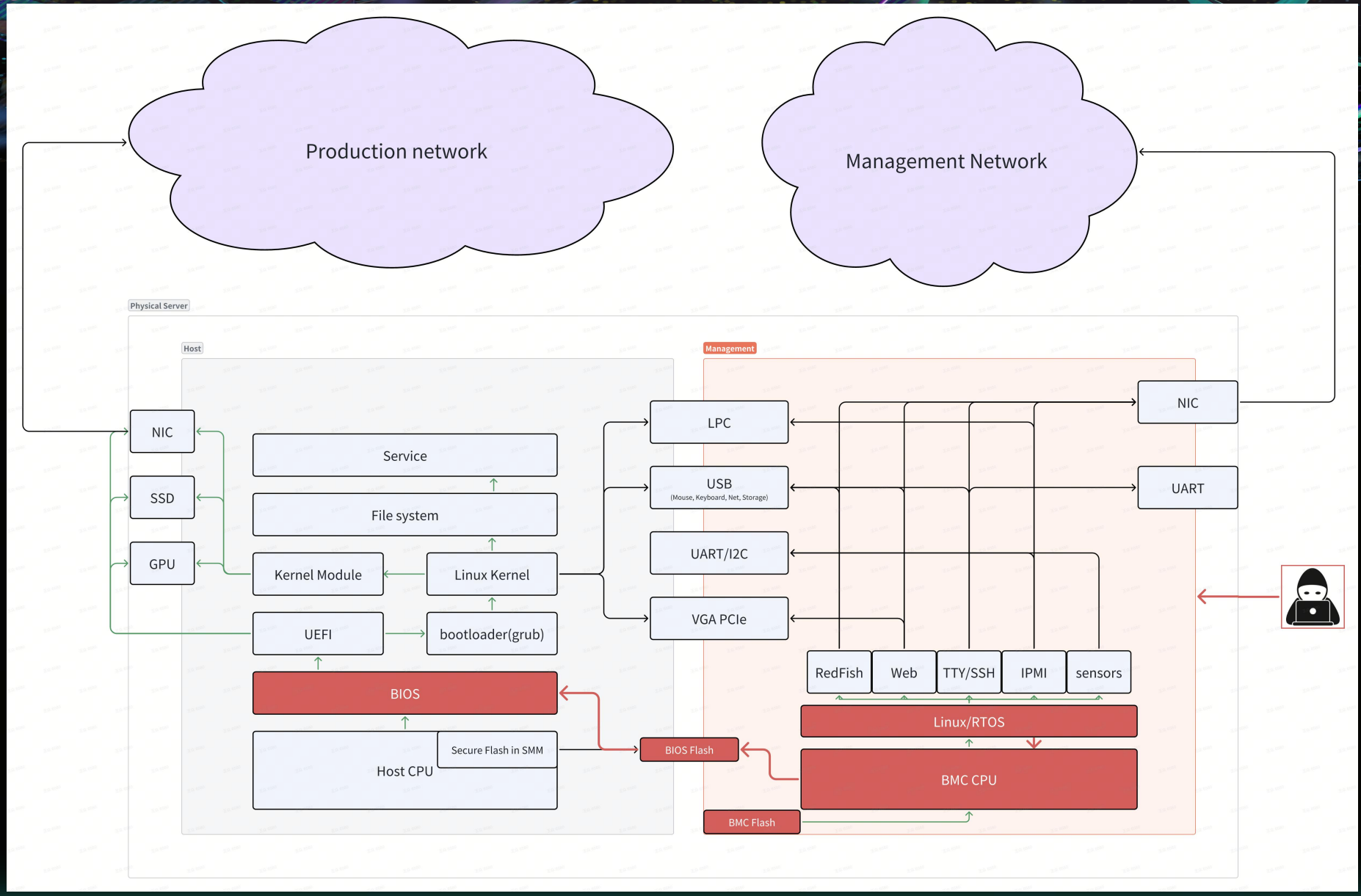
WHO HAS?

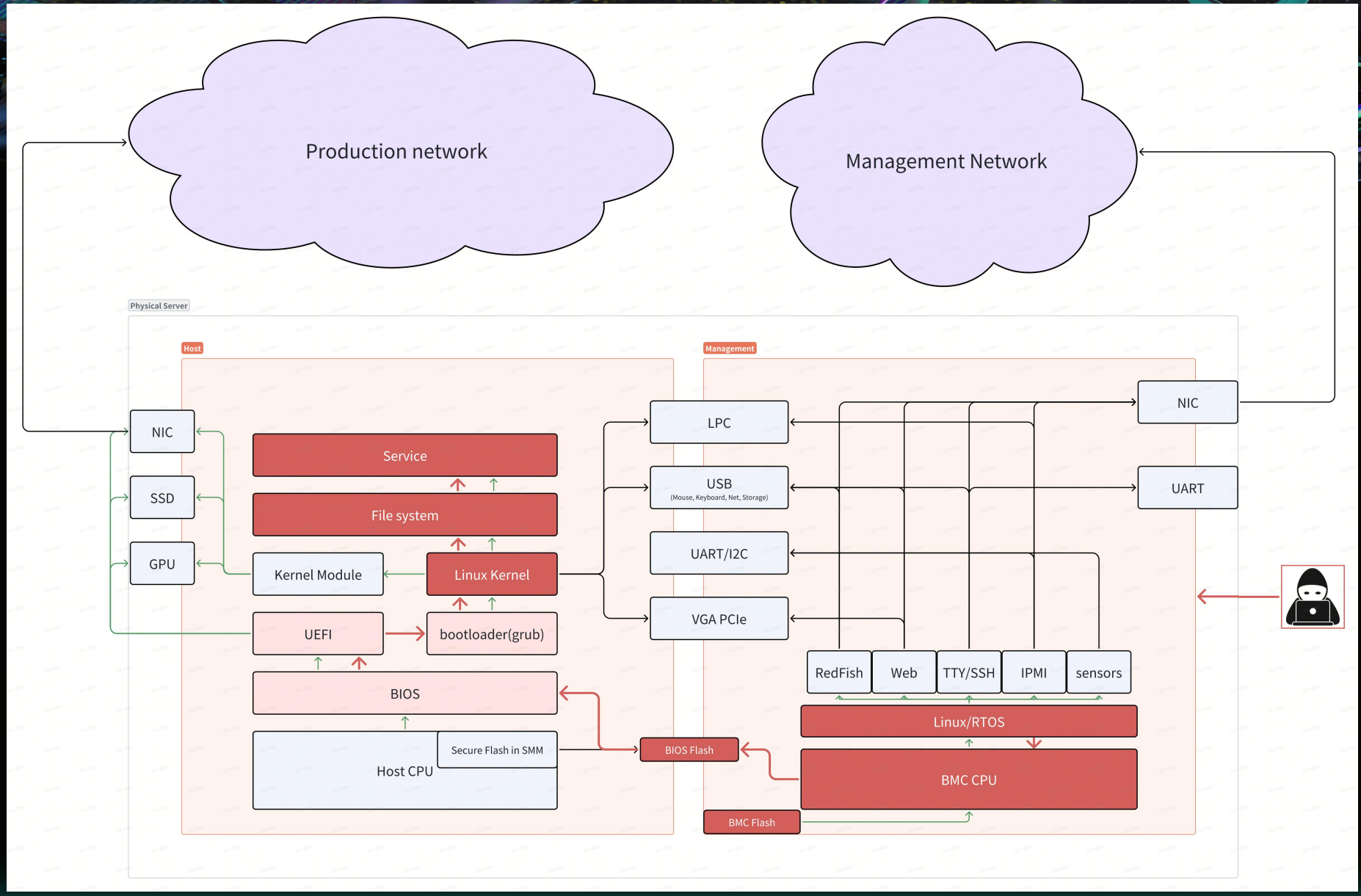












black hat ASIA 2023

MAY 11-12

BRIEFINGS

firmWar: An Imminent threat to the foundation of computing.

Vladyslav Babkin

#BHASIA @BlackHatEvents

BREAKING BMC: THE FORGOTTEN KEY TO A KINGDOM

Alex Tereshkin
Twitter: @AlexTereshkin

Adam 'pi3' Zabrocki
Twitter: @Adam_pi3

CVE-2019-6260: Gaining control of BMC from the host processor

Posted on **2019-01-23** by **Stewart Smith**

This is details for CVE-2019-6260 – which has been nicknamed "pantsdown" due to the nature of feeling that we feel that we've "caught chunks of the industry with their…" and combined with the fact that naming things is hard, so if you pick a bad name somebody would have to come up with a better one before we publish.

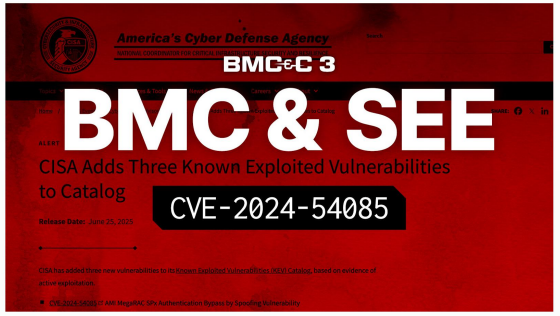
I expect OpenBMC to have a statement shortly.

The ASPEED ast2400 and ast2500 Baseboard Management Controller (BMC) hardware and firmware implement Advanced High-performance Bus (AHB) bridges, which allow arbitrary read and write access to the BMC's physical address space from the host, or from the network if the BMC console uart is attached to a serial concentrator (this is atypical for most systems).

Common configuration of the ASPEED BMC SoC's hardware features leaves it open to "remote" unauthenticated compromise from the host and from the BMC console. This stems from AHB bridges on the LPC and PCIe buses, another on the BMC console UART (hardware password protected), and the ability of the X-DMA engine to address all of the BMC's M-Bus (memory bus).

A Historic First: BMC Vulnerability CVE-2024-54085 Joins CISA's Most Critical List

By: Eclypsium | June 26, 2025



black hat ASIA 2026

WHAT HAPPENED AFTER "BREAKING BMC"

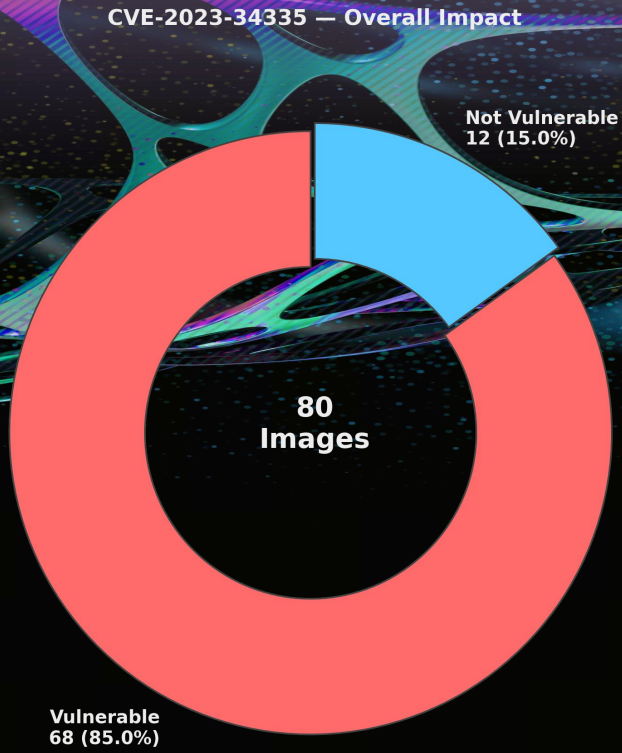
/ˈbjɑːo tʃiː/

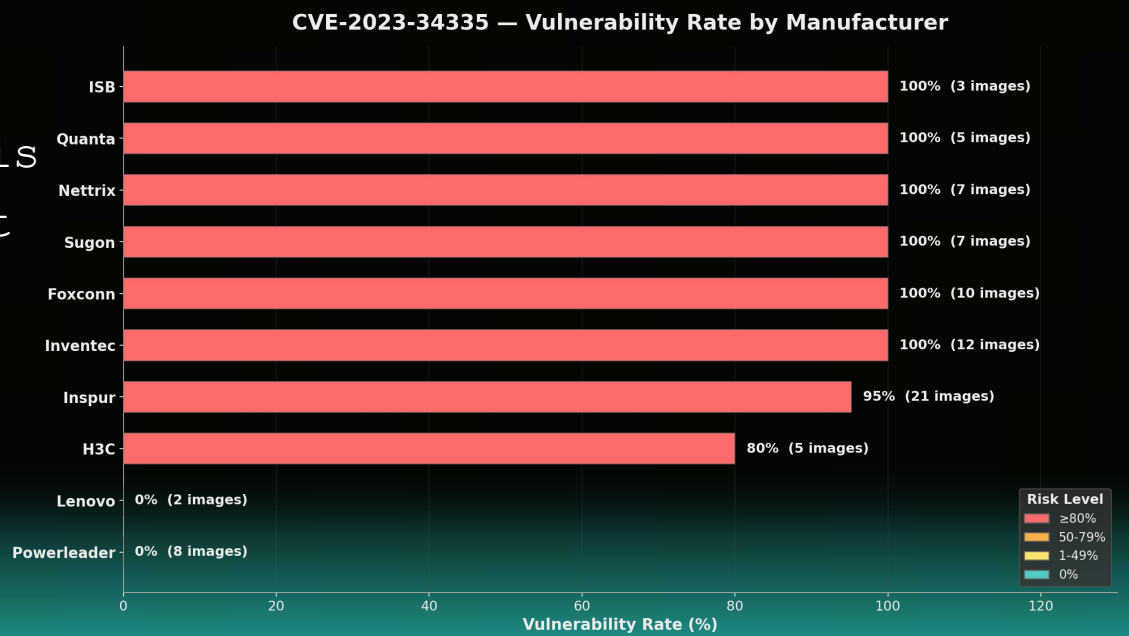
Commercial Server Manufacturer: Convenience > Security

Take CVE-2023-34335 as an example.

We analyzed 80 BMC firmwares from 10 commercial server vendor which use MegaRAC as BMC firmware solution of their products, and found that **around 85%** firmwares are still affected.

Several manufacturers stated that this "functionality" provides a convenient means for in-band firmware updates, and that the issue is addressed in **bare-metal mode customized for cloud providers**.



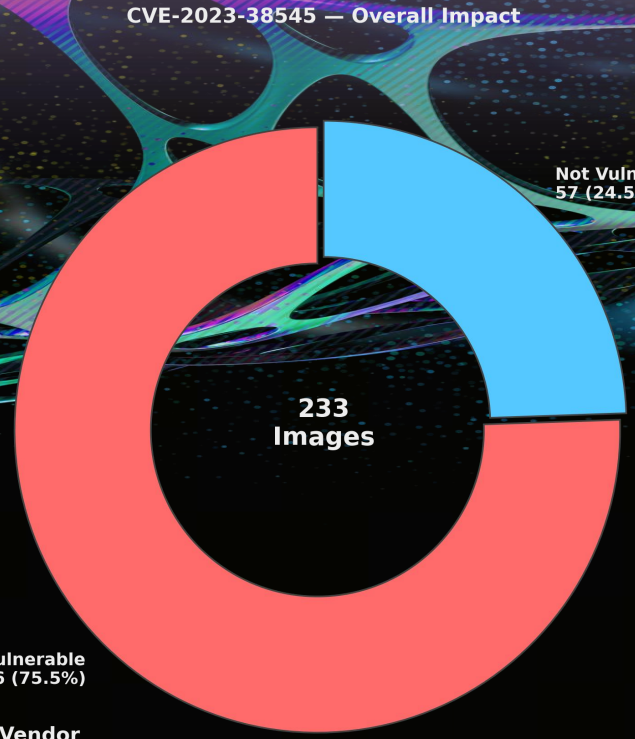


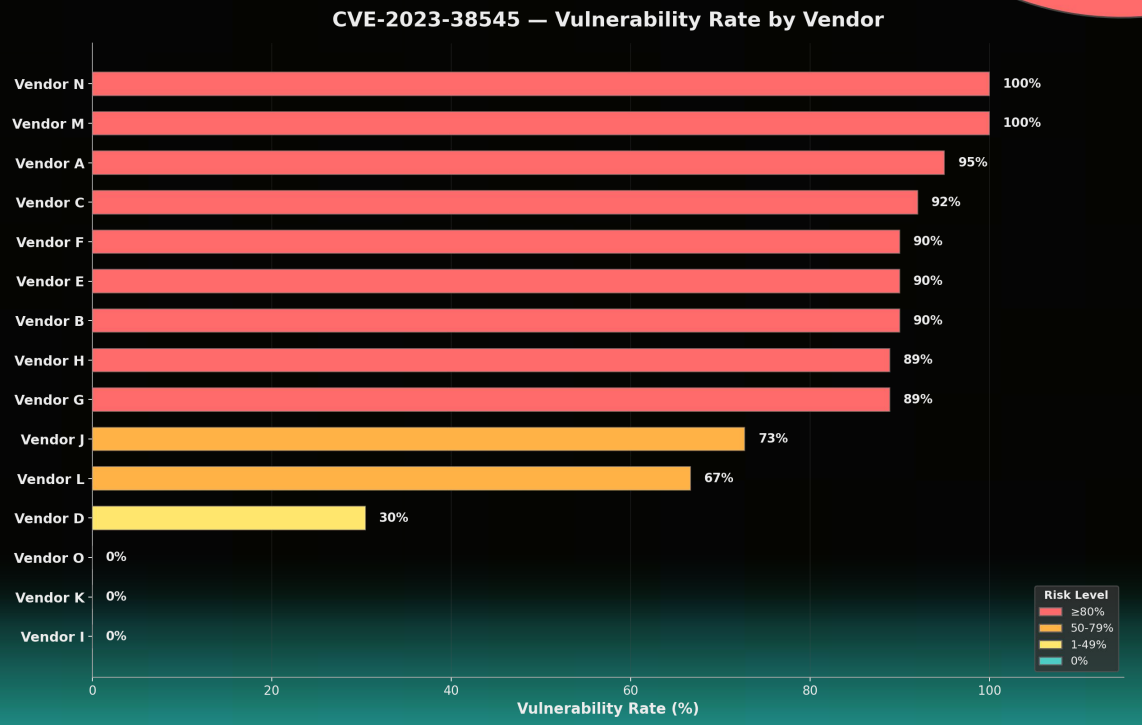
OEM/ODM Server Vendor: Igonre 3rd-party component vuln

Take CVE-2023-38545 as an example.

Cloud providers such as Google, AWS, and Aliyun usually require OxM vendors to patch known BMC vulnerabilities.

However, due to the lack of 3rd-party component vulnerability threat assessment for BMC in the industry, patches for 3rd-party component vulnerabilities are often overlooked.





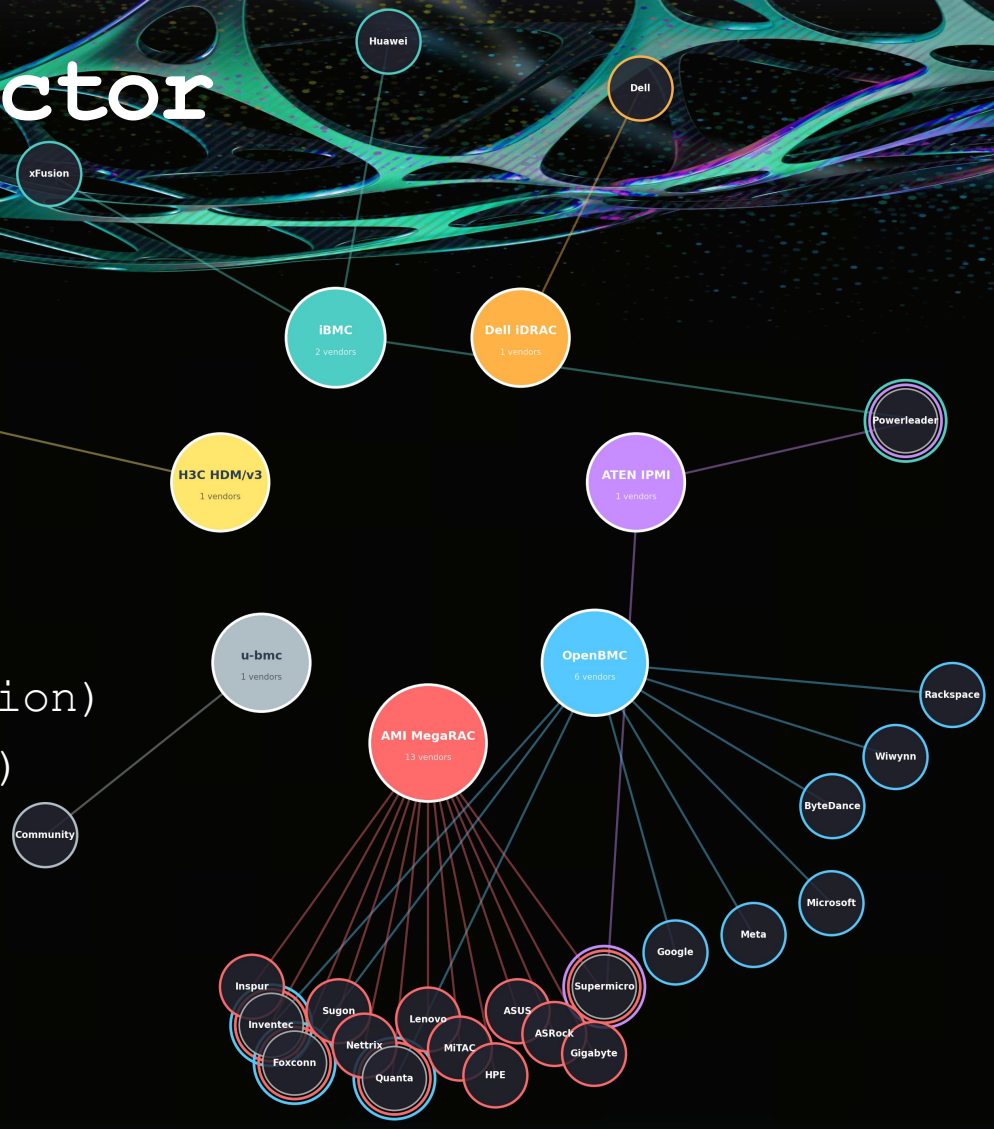

WE NEED A TOOL

Firmware Structure demo

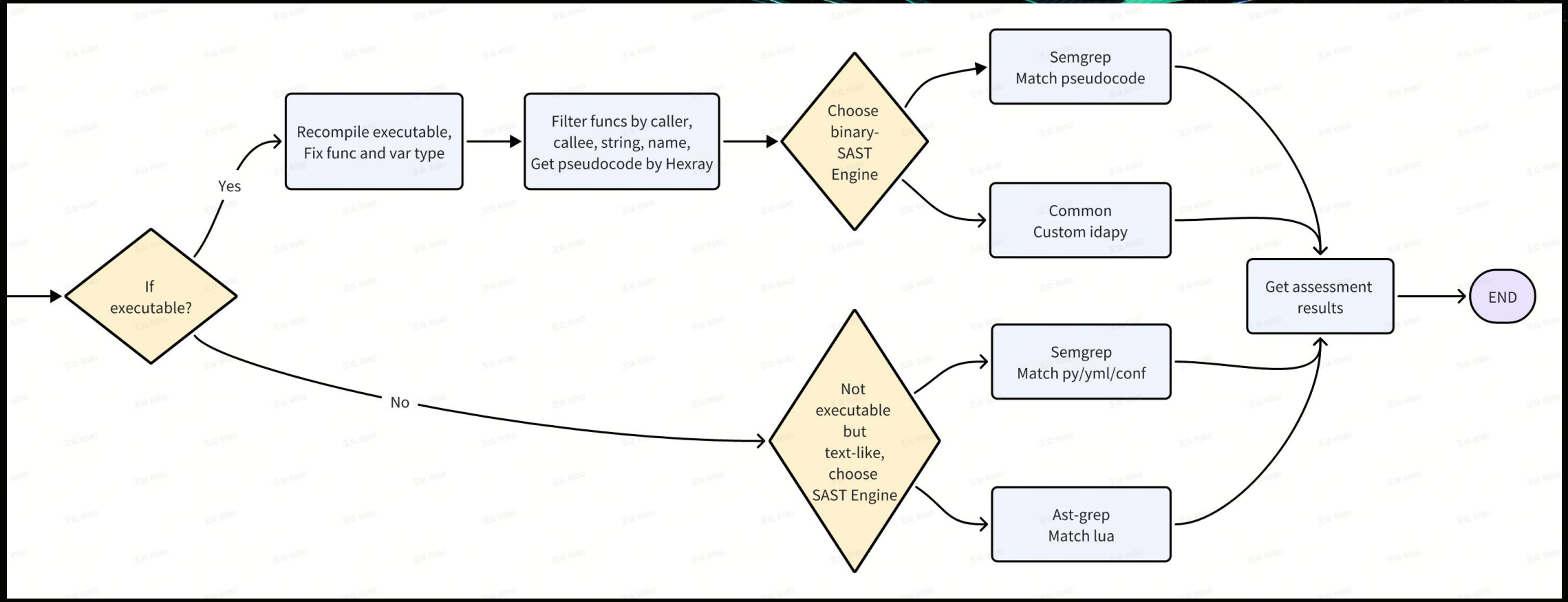
```
014:FFE0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
014:FFF0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
015:0000  24 4D 4F 44 55 4C 45 24 01 08 40 00 00 00 77 01  $MODULE$..@...w.
015:0010  00 00 15 00 00 00 00 39 72 6F 6F 74 00 00 00 00  .......9root....
015:0020  0C 03 60 00 00 00 16 00 00 30 75 01 01 00 00 00  ..`......0u.....
015:0030  00 81 4F 6C 98 00 30 30 30 30 30 30 00 00 AA 55  ..Ol˜.000000..ªU
015:0040  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
015:0050  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ

015:FFE0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
015:FFF0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
016:0000  68 73 71 73 66 14 00 00 96 CA FE 63 00 10 00 00  hsqsf...–Êþc....
016:0010  D2 01 00 00 04 00 0C 00 C0 02 01 00 04 00 00 00  Ò.......À.......
016:0020  F2 17 9C FB 00 00 00 00 6B 26 75 01 00 00 00 00  ò.œû....k&u.....
016:0030  63 26 75 01 00 00 00 00 FF FF FF FF FF FF FF FF  c&u.....ÿÿÿÿÿÿÿÿ
016:0040  0F 42 73 01 00 00 00 00 31 43 74 01 00 00 00 00  .Bs.....1Ct.....

04F:6170  00 00 00 03 00 00 00 1B 00 00 00 00 46 6C 61 74  ............Flat
04F:6180  74 65 6E 65 64 20 44 65 76 69 63 65 20 54 72 65  tened Device Tre
04F:6190  65 20 62 6C 6F 62 00 00 00 00 00 03 00 00 F3 90  e blob........ó.
04F:61A0  00 00 00 1B D0 0D FE ED 00 00 F3 90 00 00 00 38  ....Ð.þí..ó....8
04F:61B0  00 00 EB F0 00 00 00 28 00 00 00 11 00 00 00 10  ..ëð...(........
04F:61C0  00 00 00 00 00 00 07 A0 00 00 EB B8 00 00 00 00  ....... ..ë¸....
04F:61D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01  ................

04F:86D0  72 6F 66 73 40 61 30 30 30 30 30 00 00 00 00 03  rofs@a00000.....
04F:86E0  00 00 00 08 00 00 03 42 00 A0 00 00 02 50 00 00  .......B. ...P..
04F:86F0  00 00 00 03 00 00 00 05 00 00 04 0C 72 6F 66 73  ............rofs

09F:FFE0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
09F:FFF0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
0A0:0000  68 73 71 73 D1 0E 00 00 70 7F A2 5A 00 00 02 00  hsqsÑ...p.¢Z....
0A0:0010  C6 00 00 00 04 00 11 00 C0 00 05 00 04 00 00 00  Æ.......À.......
0A0:0020  CB 19 9C 7C 00 00 00 00 34 A1 04 02 00 00 00 00  Ë.œ|....4¡......
0A0:0030  2C A1 04 02 00 00 00 00 FF FF FF FF FF FF FF FF  ,¡......ÿÿÿÿÿÿÿÿ
```

Firmware Parser & Extractor

1. OpenBMC
2. ATEN(to be supported, used on supermicro motherboards)
3. MegaRAC
   I. Inspur modified
   II. Inspur encrypted
4. iDRAC
5. iBMC(to be supported, used by huawei/xFusion)
6. uBMC(to be supported, maybe a new OpenBMC)
7. HDM & HDMv3



SAST Engine



Take CVE-2023-38545 as an example

```yaml
name: CVE-2023-38545            # CVE ID
type: ["MegaRAC"]               # Target platform
hint: vuln                      # Rule type
engine: semgrep                 # Engine
detail:
  - name: CVE-2023-38545-bin        # Sub-rule name
    part: [root]                    # Filesystem to be searched
    path_pattern: \/libcurl\.so.*   # Target file path regex
    file_type: ["so", "elf"]        # File type filter
    filter:                         # Binary filtering conditions
      functions: [“funcXXX”]            # Required function names
      callers: [“callerXXX”, ...]       # Required callers
      callees: [“calleeXXX”, ...]       # Required callees
      strings: [“stringXXX”, ...]       # Required strings
    fix_func_types:
    lang: "c"                       # Decompilation language
  - name: CVE-2023-38545-conf       # Another sub-rule name
    part: [root, conf]              # Filesystem to be searched
    path_pattern: xxx.*.conf        # Config file path regex
    file_type: ["txt"]              # File type
```

board_sentinel > rules > semgrep > ! CVE-2025-23016-bin.yml

```yaml
rules:
  - id: CVE-2025-23016-vuln
    options: ⋯
    metadata: ⋯
    message: >- ⋯
    severity: ERROR
    languages: ⋯
    patterns:
      - pattern: |
          while( ... ) {
            ...
            $NAME_LEN = $GET_CHR(...);
            ...
            if ( ... ) {
              if ( $GET_STR(...) != 3 ) {
                ...
              }
              $NAME_LEN = ...;
            }
            ...
          }
      - pattern-not: | ⋯
      - focus-metavariable: $NAME_LEN
```

board_sentinel > rules > semgrep > ! CVE-2025-23016-conf.yml

```yaml
rules:
  - id: CVE-2025-23016-conf
    severity: WARNING
    languages: [generic]
    options: ⋯
    message: >- ⋯
    pattern-either:
      - pattern: |
          fastcgi.server ... ( ... =>
          ((
            ...
            "port" => ...
            ...
          )))
      - pattern: |
          fastcgi.server ... ( ...
            => (( ...
                  "port" => ...
                  ...
            ))
          ...
          )
```

Take CVE-2023-34335 as an example

The CVE-2023-34335 case requires matching data from the CmdHndlr and NetfnTbl dictionaries in MegaRAC IPMI-related shared libraries — a task clearly beyond Semgrep's capabilities.

By extension, when vulnerability detection depends on non-control-flow information, custom IDAPython scripts can be developed and integrated into BoardSentinel as plugins.

```
def check_cmd_handler_with_cmd_switch(self, display=False):
    netfn_cmd_map = {}
    for cmd_handler_item in self.cmd_handler:
        netfn = f"{int(cmd_handler_item.NetFn):02x}"
        if netfn not in netfn_cmd_map.keys():
            netfn_cmd_map[netfn] = {}
        for cmd_handler_map_item in cmd_handler_item.CmdHndlrMap:
            cmd = f"{int(cmd_handler_map_item.Cmd):02x}"
            if cmd not in netfn_cmd_map[netfn].keys():
                netfn_cmd_map[netfn][cmd] = {}
            netfn_cmd_map[netfn][cmd]["Enabled"] = "unknown"
            if isinstance(cmd_handler_map_item.CmdHndlr, func_t):
                cmd_handler_pointer = f"`{get_ea_name(cmd_handler_map_item.CmdHndlr.start_ea)}`"
            else:
                cmd_handler_pointer = f"@{int(cmd_handler_map_item.CmdHndlr)}:08x"
            netfn_cmd_map[netfn][cmd]["CmdHndlr"] = cmd_handler_pointer
    for cmd_switch_item in self.cmd_switch:
        netfn = f"{cmd_switch_item.NetFn:02x}"
        if netfn not in netfn_cmd_map.keys():
            self.debug_log(f"not found {netfn} in `cmd_handler`")
            continue
        for net_fn_cmd in cmd_switch_item.NetFunction:
            cmd = f"{int(net_fn_cmd.CmdNum):02x}"
            if cmd not in netfn_cmd_map[netfn].keys():
                self.debug_log(f"not found {cmd} under {netfn}")
                continue
            netfn_cmd_map[netfn][cmd]["Enabled"] = "false" if net_fn_cmd.Status == NetFnCmdsItemStatus.DI
    if display:
        print(json.dumps(netfn_cmd_map, indent=4))
    return netfn_cmd_map
```


BUT NOT JUST TOOLS

BMC Secure Boot

1. Trust chain:
    I. BootROM -> SPL -> U-Boot -> kernel
    II. Store Pubkey hash in OTP
    III. Enable by OTPCFG
2. Key management
    I. RSA3076/4096/ECC384
    II. Store Privkey in HSM
    III. Lockdown OTP to avoid importing another Pubkey hash
3. Version management
    I. Anti-rollback
    II. Revocation

DevSecOps

1. Classical Vulnerability
    I. CodeQL/Semgrep/...
    II. 3rd-party penetration test
    III. SBOM from SCA and BSCA
2. Threat modeling
    I. Use as a physical server or bare metal?
    II. Join production net or management net?
3. Security auditing
    I. Collect audit log and SEL of each BMC
    II. HIDS if possible
    III. Unique key per device


black hat
ASIA 2026
Q&A