Illuminating the Security Issues with Lights-Out Server Management

Anthony J. Bonkoski
J. Alex Halderman
University of Michigan
What is IPMI?

Need to manage a massive cluster of servers? OS installs, monitoring, power-cycle, etc. How?

Intel introduces Intelligent Platform Management Interface (IPMI) Specification:
- Adds a second computer
- Always on
- Integrated directly into the system buses (e.g. I²C)

OEM Names:
- HP iLo
- Dell iDrac
- Oracle iLOM
- Lenovo/IBM IMM
- SuperMicro IPMI
- ATEN IPMI
- MegaRAC
- Avocent IPMI
What is IPMI?

Baseboard Management Controller (BMC)
The embedded micro-controller: the second CPU
Typical IPMI Implementation

System
Embedded on Motherboard or Expansion card
CPU: ARM/MIPS or other low power embedded CPU
OS: Linux is common

Extra OEM Features
Remote Virtual Console
Remote Media
High network connectivity incl. HTTP and SSH.
Why do we care?

In short: IPMI is the perfect spying backdoor
Always on and often pre-enabled.
NIC failover*
Powerful Remote Tools
Widespread deployment: 100,000+ on public IPs

It’s an embedded system...
...often, security is an after-thought!

*As seen on our SuperMicro ATEN-based IPMI
Known Problems

Authentication Risks:
Many vendors ship default passwords
  root/calvin†
Anonymous undocumented accounts*
Passwords stored in plain-text*

* SuperMicro ATEN-based IPMI
† Dell iDRAC
Dan Farmer

January 2013: Starts publicly denouncing IPMI
Criticisms are largely just conjectures

Finds some negligent flaws:
- Hidden backdoor debugging web page on Dell iDRAC
- Could gain root over ssh
Is IPMI security actually a problem?
Supermicro IPMI

Supermicro SYS-5017C-LF

Nuvoton WPCM450 ARM-based BMC

IPMI Firmware by ATEN Technology

- HTML / JavaScript
- CGI (written in C)
- Linux 2.6.17
- Firmware version 1.86 (build date: 11-14-2012)
Supermicro SSH Interface

Backend: Highly modified fork of Dropbear
Frontend: Systems Management Architecture for Server Hardware Command-Line Protocol (SMASH)*

Notice: a system admin has no access to underlying Unix shell

ATEN SMASH-CLP System Management Shell, version 1.04
Copyright (c) 2008-2009 by ATEN International CO., Ltd.
All Rights Reserved

-> help
/

The managed element is the root

Verbs:
  cd
  show
  help
  version
  exit

*Distributed Management Task Force (DMTF) specification: dmtf.org/standards/smash
Reverse Engineering Approach

Fetch firmware from OEM website.
Scan and unpack: binwalk

<table>
<thead>
<tr>
<th>DECIMAL</th>
<th>HEX</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>59700</td>
<td>0xE934</td>
<td>Copyright string:</td>
</tr>
<tr>
<td>60835</td>
<td>0xEDA3</td>
<td>Copyright string:</td>
</tr>
<tr>
<td>1572864</td>
<td>0x180000</td>
<td>CramFS filesystem,</td>
</tr>
<tr>
<td>9961472</td>
<td>0x980000</td>
<td>Zip archive data,</td>
</tr>
<tr>
<td>11086483</td>
<td>0xA92A93</td>
<td>End of Zip archive</td>
</tr>
<tr>
<td>12058624</td>
<td>0xB80000</td>
<td>CramFS filesystem,</td>
</tr>
</tbody>
</table>

Mount filesystems
Objdump and IDA Pro
What to Look For?

Begin with Classics:

1. Insecure Input Validation
2. Shell Injection
3. Buffer Overflows
function PrivilegeCallback(Privilege) {
    // full access
    if (Privilege == '04') {
        isSuperUser = 1;
    }
    // only view
    else if (Privilege == '03') {
        var_save_btn.disabled = true;
    }
    // no access
    else {
        alert(lang.LANG_NOPRIVI);
    }
}

Server-side?
No permission checking.
No escaping of input passed to shell.
No string length checking in CGI.
15 of 67 CGI programs made calls to `system()`.

Confirmed shell injection in `config_date_time.cgi`:

```
Date & Time

Here you can view and modify the device's date and time

Time Zone: UTC+00:00
NTP Enable
- NTP Enable
- NTP Disable
Primary NTP Server: 127.0.0.1
Secondary NTP Server: 127.0.0.1
Date: January 15
Time: (hh:mm:ss) 13:03:50

☐ Daylight Saving Time

Refresh  Save
```
Shell Injection

15 of 67 CGI programs made calls to `system()`.

Confirmed shell injection in `config_date_time.cgi`:

```
127.0.0.1`sleep 60`
```

Getting command output

Redirect to `/nv/system_log`.

Issue GET request to `system_log.cgi`.

Create a psuedo-terminal

Wraps GET ands POST request in a python script.

```
root@localhost #
```
Buffer Overflows

Server backend:
  ... CGI programs.
  ... written in C.
  ... running as root.
Server backend: ... CGI programs. ... written in C. ... running as root.

// login.cgi
int main(void)
{
    char name[128], pwd[24];
    char *temp;
    // ... initialize ...
    temp = cgiGetVariable("name");
    strcpy(name, temp);
    temp = cgiGetVariable("pwd");
    strcpy(pwd, temp);
    // ... authenticate user ...
}
Buffer Overflows

Server backend:
... CGI programs.
... written in C.
... running as root.

// login.cgi
int main(void)
{
    char name[128], pwd[24];
    char *temp ;
    // ... initialize ...
    temp = cgiGetVariable("name");
    strcpy(name, temp);
    temp = cgiGetVariable("pwd");
    strcpy(pwd, temp);
    // ... authenticate user ...
}
Buffer Overflows

No length validation?

```html
<input name="name" size="20" maxlength="64"
```
Buffer Overflows

No length validation?

```html
<input name="name" size="20" maxlength="1000"

Please Login

Username: aaaaaaaaaaaaaaaaaaaaaa:

Password: _______________

login
```
Buffer Overflows

No length validation?

```html
<input name="name" size="20" maxlength="1000"

500 - Internal Server Error
```
Buffer Overflow Exploitability

Buffer-overflow defenses?

No DEP (Stack and Heap are executable).

No Stack Canaries.

Limited ASLR.

(Stack/Heap base addresses are randomized, but dynamic libraries are not. Return-to-libc works.)
Exploitation Challenges

Stack is randomized (ASLR).
   ...but, only 12 bits are random. Just 4096 possibilities.

We gain control on the return from main().
   Stack is small: shellcode must be compact.

BMC crashes and reboots if pounded too hard with requests.
Buffer Overflow Exploit

Solutions
Store the shell command in the name buffer.
Brute force through the stack randomization.
Limit the time between brute-force iterations.
Avg. search time: ~7 min.

Payload
Fetch (wget) and install modified SSH daemon.
Forks root shell on incorrect password.
Only 2 instructions changed!

root@localhost #
Vulnerable Models?

Cursory check of all Supermicro IPMI firmware downloads as of May 23, 2013.
   30 of 64 images appear vulnerable.
   135 device models.

Supermicro says they’re working on a fix.

Possibly affects other ATEN-based products.
The Impact

So, rooting this device is *easy*! But, what are the implications? Yet another broken embedded system?
The Impact

Only as secure as our weakest component. Entire system is now vulnerable! Adding an entire computer only weakens.
IPMI for Evil

BMC-based spyware and botnets

Rooted BMC $\rightarrow$ Rooted host system
  Mount a custom OS and reboot.

Rooted host system $\rightarrow$ Rooted BMC
  Re-flash the BMC with malicious code.

BMC rootkits
  A backdoor that survives potentially forever.

A scary thought
  IPMI meets Matrix $\rightarrow$ Is your IPMI just emulated? How do you know?
Network Measurements

Scanned all public IPs on May 7, 2013 using ZMap*. Downloaded all X.509 certs from HTTPS servers. Used identifying characteristics of default certificates.†

<table>
<thead>
<tr>
<th>Platform</th>
<th>Devices on Public IPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supermicro IPMI</td>
<td>41,545</td>
</tr>
<tr>
<td>Dell iDARC</td>
<td>40,413</td>
</tr>
<tr>
<td>HP iLO</td>
<td>23,376</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>105,334</strong></td>
</tr>
</tbody>
</table>

† Details on “identifying characteristics” may be found in our paper.
For System Operators

*Never* attach your IPMI device directly to the Internet.
   
   Use an isolated management network or VLAN.

Change default passwords and certificates.

Disable IPMI if you don’t need it.

Unfortunately: we’re at the will of the Vendor
Defenses

For IPMI Vendors

These are textbook vulns. *You have to do better.*

Apply security engineering practices.

Sign and verify firmware when flashing.

Make devices hard to deploy on public IPs.
Lessons

A Culture Clash?

Embedded

Internet

IPMI: hopefully a climax
Future Work

Analysis of other vendors’ implementations
Dell, HP, Lenovo, Oracle, etc.

Firmware update exploitation
Can an attacker inject a backdoor that persists?
Across BMC reboot? Across BMC flashes? Forever?

IPMI honeypot
Unclear whether attackers are exploiting these devices in the wild.
Some anecdotal evidence of their use as spambots.
Are they being used for other malicious purposes?
Conclusions

IPMI serves a vital role for system management.
Carries elevated risks, potential for powerful attacks.
At least some vendors are getting it badly wrong.
Farmer is correct: IPMI is a serious concern.
Our work: A call to arms.
Illuminating the Security Issues with Lights-Out Server Management

Anthony J. Bonkoski
abonkosk@umich.edu

J. Alex Halderman
jhalderm@umich.edu

University of Michigan
Zmap Scan Details

<table>
<thead>
<tr>
<th>Vendor</th>
<th>Identifying Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>SuperMicro</td>
<td>Subjects containing “<a href="mailto:linda.wu@supermicro.com">linda.wu@supermicro.com</a>” or “<a href="mailto:doris@aten.com.tw">doris@aten.com.tw</a>”</td>
</tr>
<tr>
<td>Dell</td>
<td>Subject containing iDRAC</td>
</tr>
<tr>
<td>HP</td>
<td>Subjects containing “CN=ILO” and issuers containing “iLO3 Default Issuer” or “Hewlett Packard”</td>
</tr>
</tbody>
</table>

*Landing pages spot-checked for false positives*