Opening up the Baseboard Management Controller

JESSIE FRAZELLE

IF THE CPU IS THE BRAIN OF THE BOARD, THE BMC IS THE BRAIN STEM.

In 2011 Facebook announced the Open Compute Project to form a community around open-source designs and specifications for data center hardware. Facebook shared its hardware specs, which resulted in 38 percent less energy consumption and 24 percent cost savings compared with its existing data centers.[6] What Facebook and other hyperscalers (Google, Microsoft, et al.) donate to the Open Compute Project are their solutions to the agonizing problems that come with running data centers at scale.

Since then, the project has expanded to all aspects of the open data center: baseboard management controllers (BMCs), network interface controllers (NICs), rack designs, power busbars, servers, storage, firmware, and security. This column focuses on the BMC. This is an introduction to a complicated topic; some sections just touch the surface, but the intention is to provide a full picture of the world of the open-source BMC ecosystem, starting with a brief overview of the BMC's role in a system, touching on security concerns around the BMC, and then diving into some of the projects that have developed in the open-source ecosystem.

WHAT IS THE BMC AND WHY DO WE NEED IT?

If the CPU is the brain of the board, the BMC is the brain stem. It monitors and manages the physical state of a computer or hardware device. This state includes temperature, humidity, power supply voltage, fan speeds, remote access, and operating system functions. The BMC has historically been a SuperH or ARM-based SoC (system on a chip) with common functionality including but not limited to:

- RMII (reduced media-independent interface) and RGMII (reduced gigabit media-independent interface) for ethernet.
- A boot flash with an SPI (serial peripheral interface) NOR. (NOR and NAND are types of nonvolatile flash memory; the difference is in the type of logic gate used.)
- PCI (peripheral component interconnect) express.
- An LPC (low pin count) bus for communicating with the host. Intel's successor to LPC is the eSPI (Enhanced Serial Peripheral Interface bus).

The BMC usually communicates to the outside world (or the data center control network) using the IPMI (Intelligent Platform Management Interface), a message-based, hardware-level interface specification for managing and operating computer systems. It operates independently of the operating system, the server's CPU, and the firmware that allows admins to manage a system without an operating system or any system management software. Admins can also take advantage of IPMI's local network to get a console on a remote computer that is otherwise inaccessible.

BMC SECURITY CONCERNS

The IPMI stack was not designed with security in mind (the IPMI spec requires making the hash of a user's password available over the stack). The assumption was that the data center control networks would be segregated and trusted, which is why IPMI is notorious for security vulnerabilities.[2] Exploits in the IPMI stack and the BMC are devastating because of the many privileged operations for which they are responsible. Improving IPMI security has historically been neglected, as most IPMI software is proprietary.

Exploits in the IPMI stack and the BMC are devastating because of the many privileged operations for which they are responsible.

The BMC has its own problems with largely proprietary software and vulnerabilities. The most recent notable BMC vulnerability is USBAnywhere,[3] discovered by Rick Altherr, principal engineer at Eclypsium. On Supermicro servers, an attacker can use USBAnywhere to connect remotely to a server and virtually mount any USB device to the server. As a result, an attacker could load a new operating system image or implant a firmware backdoor to facilitate ongoing remote access. At the time of the disclosure, 47,000 vulnerable systems were found to be exposed to the public Internet. Another fun vulnerability is Pantsdown,[1] which allows read and write access to the BMC's address space from the host. Pantsdown is an example of a requested feature causing a vulnerability.

But wait, it gets worse. As Trammell Hudson pointed out in his Modchips of the State talk at the 35th Chaos Communication Congress in 2018,[7] the BMC often has access to the host firmware via SPI (serial peripheral interface) and to host memory through DMA (direct memory access). The BMC gets DMA access because it is on the PCIe (peripheral component interconnect express)

bus as a device. This means it can inject code into the host's firmware. Much BMC firmware also lacks the notion of a secure boot. This makes the BMC a prime target for hackers. Here I emphasize a point I made in a previous article on open-source firmware:[5] It's an alarming problem that the code running with the most privilege has the least visibility and inspectability.

THE BMC BECOMES OPEN SOURCE

The trend toward open sourcing the data center has led to a number of innovative BMC projects.

OpenBMC

In 2014 Facebook decided to solve the problems with proprietary BMC software by starting an open-source BMC software project.[4] In 2015 IBM and Rackspace collaborated on solving the same problems with their own project.[9] Both projects were called OpenBMC and ended up merging into the OpenBMC project the firmware community is familiar with today (https://github.com/openbmc/openbmc). The founding organizations of the OpenBMC project, post-merger, were Microsoft, Intel, IBM, Google, and Facebook. OpenBMC has the widest range of support for various BMCs.

The OpenBMC project encompasses u-boot, an open-source bootloader that boots a Linux kernel with a minimal root file system containing all the tools and binaries needed to run OpenBMC. OpenBMC is designed with a service-oriented approach. Services are started and maintained by systemd and communicate with each other over dbus. Designing for services makes sense as an easy

way for multiple collaborators and vendors to contribute to a single BMC implementation. This allows each vendor contributing to the codebase to have separate daemons it can turn on to ship in its specific distribution of OpenBMC; however, it also makes the BMC software more complex to debug, audit, and put into production.

u-bmc

After OpenBMC came u-bmc (https://github.com/u-root/u-bmc), a software project started by Christian Svensson of Google. Written in Go, u-bmc aims for a more minimal BMC software architecture, challenging the status quo by replacing IPMI with gRPC. Removing IPMI makes u-bmc provocative from a security perspective since the attack surface area is reduced. Unlike OpenBMC, u-bmc boots a Linux kernel directly from the ASPEED startup code after DRAM initialization, thus removing the need for a bootloader such as u-boot. As of the publication of this article, u-bmc supports BMCs based on the ASPEED AST2400 and AST2500, but plans to support more in the future and always welcomes contributions. If you have a Supermicro X11SSH board that supports coreboot, it is possible to use u-bmc as your BMC software.

RunBMC

Not only has software around the BMC been open sourced, but the hardware has as well. Eric Shobe and Jared Mednick of Dropbox analyzed all the BMC system topologies and their differences on a platform-by-platform basis. The result was RunBMC, a standard hardware interface for BMCs. Dropbox donated version 1 of the

RunBMC hardware specs, along with two reference boards for the Nuvoton NPCM750R and ASPEED 2500 RunBMC modules, to the Open Compute Project in August 2019.[8]

The RunBMC design allows for swapping out BMCs separate from the rest of the board, isolating and locking down the BMC subsystem. Previous to this, the BMC was soldered onto the board. This is compelling from a security perspective since focus is shifted to a single, swappable BMC card, which can easily be replaced if broken, updated with a different version, or integrated with other security features. For example, a root of trust, the trusted source that verifies system software before execution, can secure I/O between the BMC card and the rest of the board. This also allows users to switch easily between the common BMC manufacturers, ASPEED and Nuvoton. Interesting fact: Sun also had a BMC interconnect with its ILOM (Integrated Lights Out Manager), as did Dell with DRAC (Dell Remote Access Controller), HP with iLO (Integrated Lights-out), and IBM and Lenovo with IMM (Integrated Management Module)—however, most don't ship this way today.

Open sourcing the software at the lowest levels of the stack provides visibility into the code that is running with the most privileges on systems.

OPEN SOURCE MOVING THE ECOSYSTEM FURTHER

OpenBMC set the stage for BMC firmware and hardware to be open sourced. This spawned a series of other innovations being open sourced, and more can be expected. This space is turning out many awesome projects, and I am lucky to be able to shine a light on the amazing work being done. Open sourcing the software at the lowest levels of the stack provides visibility into the code that is running with the most privileges on systems. We can only hope

Related articles

Security for the Modern Age
Securely running processes that require the entire syscall interface
Jessie Frazelle
https://queue.acm.org/detail.cfm?id=3301253

Commercializing Open Source Software
Many have tried, a few are succeeding, but challenges abound.
Michael J. Karels
https://queue.acm.org/detail.cfm?id=945125

GNL is Not Linux
What's in a name?
Kode Vicious
https://queue.acm.org/detail.cfm?id=2909572

that this will lead to more eyes vetting the code, encourage more minimal architectures, and lessen the risk of systems being caught with their "Pantsdown" in the future.

Acknowledgments

Huge thanks to the individuals in the open-source ecosystem for helping me learn about their projects: Rick Altherr, Chris Koch, Christian Svensson, Ron Minnich, Trammell Hudson, Eric Shobe, and Jared Mednick. Looking forward to the future of open-source firmware. If you are interested in helping with any of the projects mentioned here, check out GitHub.

References

1. Common Vulnerabilities and Exposures. CVE-2019-6260; https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-6260.
2. Common Vulnerabilities and Exposures. Intelligent Platform Management Interface; https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=ipmi.
3. Eclypsium. 2019. Virtual media vulnerability in BMC opens servers to remote attack; https://eclypsium.com/2019/09/03/usbanywhere-bmc-vulnerability-opens-servers-to-remote-attack/.

4. Fang, T. 2015. Introducing "OpenBMC": an open software framework for next-generation system management. Facebook Engineering; https://engineering.fb.com/open-source/introducing-openbmc-an-open-software-framework-for-next-generation-system-management/.
5. Frazelle, J. 2019. Open-source firmware. *acmqueue* 17(3); https://queue.acm.org/detail.cfm?id=3349301.
6. Heiliger, J. 2011. Building efficient data centers with the Open Compute Project. Facebook; https://www.facebook.com/notes/facebook-engineering/building-efficient-data-centers-with-the-open-compute-project/10150144039563920/.
7. Hudson, T. 2019. Modchips of the State; https://trmm.net/Modchips#Defenses.
8. Shobe, E., Mednick, J. 2019. RunBMC: OCP hardware spec solves data center BMC pain points; https://blogs.dropbox.com/tech/2019/08/runbmc-ocp-hardware-spec-solves-data-center-bmc-pain-points/
9. Sullivan, A. 2015. OpenPOWER & Open Compute: full speed ahead with Barreleye; https://blog.rackspace.com/openpower-open-compute-barreleye.

Jessie Frazelle *is an independent consultant. She previously worked on the Docker Core Team, followed by Google and Microsoft. She loves computers and diving deeper into the various layers of the stack. This is just one of her adventures into learning about something new.*