Professor: Dr. Brown

Emerging Technologies Within Hardware Security

COM-203-OL02

Jyothi Cameron
6-20-2021

Author Foreword

At a very young age, I was interested in robotics. The transition from the ephemeral concept of "technology" to a tangible machine that can produce work was fascinating to me. This fascination abstracted into a curiosity with computers, then evolved into an interest in both computer hardware and software, aptly named "computer engineering" and "computer science" respectively. However, there was a missing link for me. *How did these systems "speak" to one another?* In pursuit of this I discovered system development; kernels and the like. The complexity intimidated me, and I laid rest of those pursuits for a time being until I had another question: *if I learned how this works, what could I make a computer do?*

In other words, I was asking myself how I could exploit a computer's systems.

With age, I discovered the sort of "Wild West" place the internet was, and still is. News pervades the headlines of hacking attempts whether successful or not, and potential data breaches. At every level, it seemed sensitive data was vulnerable to compromise and perennially subject to the efforts of malicious adversaries, until I discovered the information security. Not so bleak was the situation as I had assumed: people *much* smarter and *much* more talented than I were working tirelessly to combat these adversaries on all fronts, whether it be physically or digitally. It was their business to put these guys "out of business."

I admired them and wanted to be like them; I was inspired. That led me to Saint Leo University's cybersecurity program.

I am not so oblivious that I am unaware that technology is advancing even as I take time to learn it, there is a saying I heard many times ago remarking on the speed of information technology today, saying to the effect that "once something had become 'standard,' it was already obsolete." Because of this I took it upon myself to keep a pulse on current events, leading me to discover that leaps and bounds had been made to secure computers.

Here, I introduce to you the Morpheus Processor, under the assumption that the reader is already familiar with the basics of computers.

Background

To address the successes and concerns regarding this emerging technology, we must first take a historical look at computer hardware and the prevalence of current architecture.

A computer is at its heart a mathematical tool. Instruction sets in the form of "bits" are fed into microchip logic gates. Under the basis that a 1 is "TRUE" and a 0 is "FALSE," we can build logical operations using combinations of different gates and their outputs. This was an important design choice because it was an extremely simple foundation that was also scalable. This "scaling" is how developers today can abstract away from that low-level machine code and use extremely high-level languages like Python, which at times can look syntactically synonymous with written English. This ease of use has allowed computers to invade every aspect of the world, and as such required more and more data to perform more and more tasks – because remember, a computer is a mathematical tool. This has become a double-edged sword in today's always-on, always-connected, internet-of-things ecosystem.

Not only is a computer a mathematical tool, but it is also one built by humans, who are inherently flawed. This leads to every aspect of computers containing some flaw that could be exploited by malicious actors. Naturally, the scope of a computer's tasks (ranging from desktop and multi-server cloud solutions) depends entirely on subjective use cases. The average desktop computer user will not be running homebrew multi-core algorithms to shave milliseconds from a programs clock speed, and likewise a large corporation will not use the cumulative power of their data centers to store grandma's cat pictures (although this could unironically be said for the likes of Drobox and Carbonite, but these are services tailored specifically for that use case and lie outside of this generalized example). Because the *needs* are different, the *risks* are consequently different, and thus the *solutions* must be tailored to use cases and are sometimes extremely subjective (and expensive). While a single person may suffice with a single flash drive for data safeguarding, a large corporation does not have that luxury – their servers need to maintain constant uptime and offer data redundancy in case of loss. The difference between the two is several million dollars **(Stream Data Centers, 2020)**.

Approximately 96.3% of the world's top 1 million servers run on Linux, and 90% of all cloud infrastructure operates on Linux, while over 80% of all smartphones are based on it **(Galov, 2021)**.

Meltdown targeted windows and Linux on 64-bit intel processors, along with some ARM-based microprocessors. Assuming only scope of "Linux servers on 64-bit Intel processors and Android mobile devices on ARM processors," using the statistics above it is not hard to assume the dramatic impact this exploit had on both personal and server solutions worldwide, especially considering that not only is the number of mobile devices growing **(O'Dea, 2020)**, many small business server solutions include using Linux on a Raspberry Pi, which also uses ARM architecture. This pervasiveness could not be overlooked, and the fascinating part about it to me was that this was a hardware issue – the problem could only truly be fixed by manually changing hardware and not simply patching over it with software. This was demonstrated with Intel's scramble to develop the Cascade Lake processors that were released only months after the exploit was made public (**Shankland, 2018)**.

Why was something as destructive as this created, though? The most succinct answer is that it is profitable. If success is based on how many machines one can compromise, then it is in one's best interest to compromise as many machines as possible. From an adversarial standpoint, this is how they think. It is far more profitable to make malware for windows and Linux machines simply because there are more of them, and by extension it is even more profitable to compromise intel and ARM processors, as those types corner the market by a large margin.

This subsequently began my search into architecture-based information security techniques, and how they differentiated and/or improved upon my knowledge of processors at the time.

Hardware Security and Morpheus

What is "hardware security"? What does it entail? In the paper Hardware Security written by Swarup Bhunia and Mohammad "Mark" Tehranipoor, hardware security is managed by "primitives," or the actual physical devices responsible for securing data, described as:

"[playing] an important role in ensuring trust, integrity, and authenticity of integrated circuits (ICs) and electronic systems. Primitives, such as physical unclonable functions (PUFs) and true random number generators (TRNGs) produce device-intrinsic electronic fingerprints and random digital signatures, respectively, to generate cryptographic keys and IDs commonly used for device authentication, cloning prevention, generating session keys, nonce, and many more." **(Bhunia and Tehranipoor, 2019).**

With that in mind, these primitives are supremely sensitive. What Meltdown attempted to do in essence was compromise memory isolation, or the ability for kernel address ranges marked for privileged access only, to be accessed by a user or program thus subverting hardware-level encoding for privilege checking **(Lipp et. al, 2020, p1).** This is aligned with what is further described in Hardware Security chapter 1.5.1 "Attack Vectors":

> *"Attack vectors—as they relate to hardware security—are means or paths for bad actors (attackers) to get access to hardware components for malicious purposes, for example, to compromise it or extract secret assets stored in hardware. Example of hardware attack vectors are side-channel attacks, Trojan attacks, IP piracy, and PCB tampering. Attack vectors enable an attacker to exploit implementation level issues (such as, side-channel attacks and PCB tampering) or take advantage of lack of control on hardware production cycle (such as, Trojan attacks)."* **(Bhunia and Tehranipoor, 2019).**

As a knee-jerk reaction to Meltdown (and its cousin Spectre), patches were implemented that no longer mapped kernels in the user space, thus making the exploit obsolete **(Lipp et. al, 2020, p11).** This was only a temporary fix, though, and soon more concrete solutions had to be created. Physical ones. The entirety of the 64 bit architecture had to change. To prevent redundancy, I will omit commercial solutions and proceed to experimental research; i.e. Morpheus.

In the summer of 2020, 580 cybersecurity researchers spent 13,000 hours trying to break into a new kind of processor. They all failed **(Moore, 2021)**. It is the only processor to date within the DARPA "Security Integrated Through Hardware and firmware" (SSITH) program that holds this record.

It works in essence like a Rubik’s Cube, changing its underlying “undefined semantics” as university professor Todd Austin says, every few hundred milliseconds. This sounds abstract, so here is some math; an excerpt from xxx:

“The original (3 × 3 × 3) Rubik’s Cube has 8 corners and 12 edges. Corners can be arranged in 8! (40,320) ways, and there are 3^7 *(2187) possible orientations because the orientation of the eighth (final) corner depends on the preceding ones. Edges can be arranged by 12!/2 (239,500,800) ways, restricted from 12! because edges must be in an even permutation exactly when the corners are. There are* 2^{11} *(2048) possibilities because 11 edges can be flipped independently, with the flip of the 12th depending on the preceding ones. The number of Rubik’s Cube configuration species is about*

8!×37×(12!/2)×211=43252003274489856000

which is approximately 43 quintillion.” **(Zeng et. al., 2018)**

For comparison, professor Austin says they used 200 knobs for this iteration, thus resulting in 2^200 (or two states, 1 and 0, raised to the power “n” which is the number of knobs used), which is 1606938044258990275541962092341162602522202993782792835301376 possible states.

It seems neigh impenetrable, but there are drawbacks. For one, the processor is about 10% slower than other processors, and fairly expensive to manufacture due to overheads incurred from its custom design. This leads into my next discussion.

Concerns

Before addressing the technology, there is the issue of this solution's price, performance cost, complexity and subsequent integration at scale.

There is no doubt that for what it does, it does extremely well – but as it stands Morpheus cannot be produced at scale with a reasonable profit margin. The average consumer may see the $250 price tag of the current Intel Core i5 at time of writing and scoff at it; there is not a genuine desire for a consumer to pay a premium for a niche product that, inherently, exemplifies safeguards that should be present in all processors to begin with (a challenge the layman may underestimate and thus take for granted). In fact, niche products are one of the leading causes of hardware-based security breaches according to TechTarget:

> *"Custom chipsets continue to anchor a great deal of the hardware within corporate data centers or in high-end desktops. Because these purpose-built chips are tailored for niche purposes, manufacturer security reviews are not nearly as intense as those conducted for chips that are to be installed in much larger groups of devices."* **(Froehlich, 2020)**

While this seems contradictory to the previous statement that a large number of security experts were unable to crack this after 13,000 man-hours, it does not take into account advancements in technology (i.e. faster processing power) that may overcome it. Professor Austin himself says he wishes to decrease the time that Morpheus shuffles itself to as little as 10 milliseconds – a near 10x decrease. This means more research, and subsequently higher costs. A fact I omitted earlier was that Morpheus ran on not only a unique chipset, but also a unique *operating system* as well, which will most likely slow it down due further to bloat. It is a proprietary turnkey system at its heart and adding more complexity to the hardware will mean adding complexity to the software as well. That brings me to the problem of integrating something like this. Edutopia lists three core ideas about successful integration:

> “*Successful technology integration is achieved when the use of technology is:*
>
> - *Routine and transparent*
> - *Accessible and readily available for the task at hand*
> - *Supporting the curricular goals, and helping the students to effectively reach their goals*”
>
> **(Edutopia, 2007).**

Niche, proprietary hardware is the antithesis to both “routine and transparent” (in terms of maintenance) and “accessible and readily available” (per its prohibitive cost and steep learning curve). As such, by this definition it is not suitable for integration at scale. However, there is a rebuttal: data centers. They are *supposed* to be resilient to attack, and the organizations that employ them can certainly foot the bill after analyzing the cost to benefit ratio. It certainly is not for the average consumer, but for critical infrastructure like a data center with far-reaching scope and catastrophic implications upon failure, something like this can be argued as not only vital but necessary.

Conclusions

Conjecture aside, I am just one novice student, and my deliberations can be taken with a grain of salt. I am by no means an expert on this subject matter and cannot necessarily attest to the veracity of these studies – but my references are from reputable sources, in many instances firsthand accounts. With more experience I can possibly make more accurate and meaningful inferences from these studies and their possible implications, but for now I can state facts and hope my generalizations are not too far off. Saint Leo's Core value of Integrity is "*The commitment of Saint Leo University to excellence demands that its members live its [mission] and deliver on its promise. The faculty, staff, and students pledge to be honest, just, and consistent in word and deed."* In terms of computer science, this directly speaks to so called "hacker ethics." Without going too far into detail, it can be summarized with Uncle Ben's quote to Peter Parker in the Spider-Man series: "with great power comes great responsibility." As an aspiring information security specialist, it would be my job to research emerging technologies such as this and make predictions about its use cases as I have done in the above sections, so that I may or may not decide to invest time learning a new skill set to properly incorporate this new technology into my corporation's current business structure.

My interest in computer security whether specifically hardware or software will never wane, and whether it is sanctioned or by personal interest I will continue to stay in tune with matters regarding emerging technologies, like microprocessors being made from folded strips of graphene **(E&T, 2021)** or new LiFi technology.

Sources

Bhunia, S., & Tehranipoor, M. (2019). Abstract. In *Hardware security: a hands-on learning approach* (pp. 311–345). essay, Elsevier, MK, Morgan Kaufmann Publishers.

Bhunia, S., & Tehranipoor, M. (2019). 1.5.1 Attack Vectors. In *Hardware security: a hands-on learning approach* (pp. 1–20). essay, Elsevier, MK, Morgan Kaufmann Publishers.

E&T editorial staff. (2021, February 15). *Smallest microchips yet folded from graphene*. The Institution of Engineering and Technology. https://eandt.theiet.org/content/articles/2021/02/smallest-microchips-yet-folded-from-graphene/.

Edutopia. (2007, November 5). *What Is Successful Technology Integration?* Edutopia: Technology Integration. https://www.edutopia.org/technology-integration-guide-description.

Froehlich, A. (2020, November 20). *What are the biggest hardware security threats?* SearchSecurity. https://searchsecurity.techtarget.com/tip/What-are-the-biggest-hardware-security-threats.

Galov, N. (2021, January 16). *111+ Mind-boggling Linux Statistics and Facts for 2021 - Linux Rocks!* HostingTribunal. https://hostingtribunal.com/blog/linux-statistics/#gref.

Lipp, M., Schwarz, M., Gruss, D., Prescher, T., Haas, W., Horn, J., Mangard, S., Kocher, P., Genkin, D., Yarom, Y., Hamburg, M., & Strackx, R. (2020). Meltdown. *Communications of the ACM*, *63*(6), 46–56. https://doi.org/10.1145/3357033

Moore, S. K. (2021, April 13). *Morpheus Turns a CPU Into a Rubik's Cube to Defeat Hackers*. IEEE Spectrum: Technology, Engineering, and Science News. https://spectrum.ieee.org/tech-talk/semiconductors/processors/morpheus-turns-a-cpu-into-a-rubiks-cube-to-defeat-hackers.

O'Dea, S. (2020, December 18). *Number of mobile devices worldwide 2020-2024*. Statista. https://www.statista.com/statistics/245501/multiple-mobile-device-ownership-worldwide/.

Shankland, S. (2018, March 15). *Your data will get safer later this year when Intel Xeon chips start blocking Spectre attacks*. CNET. https://www.cnet.com/news/intel-blocks-spectre-attacks-with-new-server-chips-this-year/.

Stream Data Centers. (2020, August 26). *Data Center Cost*. Stream Data Centers. https://www.streamdatacenters.com/glossary/data-center-cost/.

Zeng, D.-X., Li, M., Wang, J.-J., Hou, Y.-L., Lu, W.-J., & Huang, Z. (2018, August 27). *Overview of Rubik's Cube and Reflections on Its Application in Mechanism*. Chinese Journal of Mechanical Engineering. https://cjme.springeropen.com/articles/10.1186/s10033-018-0269-7.