Are Cold Boot Attacks Still Feasible: A Case Study on Raspberry Pi With Stacked Memory

Yoo-Seung Won and Shivam Bhasin
Temasek Laboratories, Nanyang Technological University, Singapore
{yooseung.won, sbhasin}@ntu.edu.sg

***Abstract*—Cold boot attacks are semi-invasive attacks which have threatened computer systems over a decade now to leak sensitive user information passwords, keys and PIN. With internet of things (IoT) finding mass deployment, their security must be well investigated. In this work, we take a look at popular IoT device Raspberry Pi (model B+), which is already deployed in millions. Raspberry Pi features a stacked memory on top of its processor, making it impossible to physically separate the RAM from the processor. We investigate the decay model of a cold boot attack on Raspberry Pi. The results show a decay rate as low as 0.00027% which is orders of magnitude lower than previous works allowing close to perfect data recovery. We further report successful recovery of secret disk encryption key when using dm-crypt on Raspberry Pi followed by discussion on mitigation strategies.**

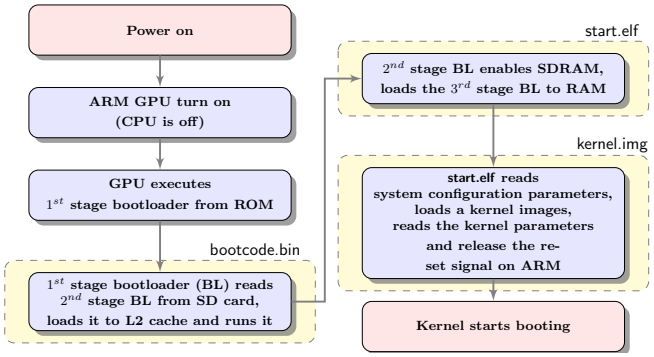

Fig. 1. Procedure of boot process on Raspberry Pi model B+

I. Introduction

Cold boot attacks are known over a decade now, making sensitive RAM content vulnerable. In 2008, Halderman *et al.* [1] showed that the RAM data which may contain sensitive information like passwords, keys *etc* can be recovered, while keeping RAM at temperatures as low as -50°C. Unlike normal temperature, the RAM data may persist several minutes at low temperatures in the memory, allowing easy recovery. The feasibility of cold boot attacks was also later shown on smart phones such as Galaxy Nexus [2] and even advanced memory technologies such as DDR3 and DDR4 RAM which have memory scrambler countermeasure [3], [4]. Hackers have also demonstrated cold boot attacks in sleep mode on a Windows PC by manipulating security settings in the boot sequence to disable initialization [5].

With the passing decade, technology has changed a lot. Internet of things (IoT) have found wide application in edge-oriented use cases like smart home, smart factory *etc*. Edge based deployment raises new security concerns [6]. The security issues span beyond traditional OS level to device level owing to easy access at the edge. The low-cost constraint on edge devices leaves little room for protection layers. Although the devices used in IoT are low-cost, they do come in complex packaging adopting methods like stacked memory, package on package *etc*. Won *et al.* [7] presented a first vulnerability investigation of modern IoT devices with stacked memory against cold boot attacks and reported high recovery rate from stacked RAM. While [7] explores the feasibility of cold boot attacks, the work is done in simple setting with no encryption enabled.

In this work, we investigate practical cold boot attacks to recover encryption keys from IoT devices. We consider Raspberry Pi (model B+) as our target IoT device similar to [7] and being one of the widely used IoT device deployed in millions of units. This device uses a ROM based boot sequence which involves level 2 (L2) cache and SDRAM as the memory. It is known that the first stage initialization involving L2 cache is initialized [7], [8] but the later stages of boot sequence which handle SDRAM do not initialize it, opening a window of exploitation. We develop an attack to demonstrate practical encryption key recovery from Raspberry Pi. The presented attack is extremely low-cost and the net cost of required tools is below $10. The main contributions of this work are:

- We implement a cold boot attack on Raspberry Pi to analyze the underlying decay model. Our results report orders of magnitude higher recovery rate than previous cold boot attacks as in [1].
- We demonstrate a practical secret encryption key recovery attack on disk encryption on Raspberry Pi using dm-crypt.
- Finally, we discuss solutions to mitigate such vulnerabilities.

II. Cold Boot Attack On Raspberry Pi

In this section, we recall the boot sequence of target board, adversary assumption and our experimental environment to perform cold boot attack as previously described in [7].

A. Boot Sequence of the Target IoT Device

To better understand the feasibility of cold boot attack on our target board Raspberry Pi model B+ (hereafter called Pi

for brevity), we study the boot sequence of the device. In particular, we need to determine if the boot sequence initializes the RAM or not. Failure to initialize the RAM makes the target vulnerable to cold boot attacks.

The boot sequence which is illustrated in Figure 1, executes as follows. The ARM GPU is turned on before the CPU at power-up. The first stage boot loader (FSBL) is then executed by the GPU directly from the ROM. The presence of the boot code in the ROM ensures no malicious modification. The FSBL loads second stage boot loader from SD card to L2 cache. The next step is to load the third stage boot loader to SDRAM from the SD card. When the second stage boot loader enables SDRAM while its execution, the handle is passed to third stage boot loader (loaded to SDRAM from SD card) for execution. The last step (optionally) reads config.txt for system configuration through start.elf and finally executes kernel.img.

B. Adversary Assumption

The main objective of the adversary for the following attack is to recover the RAM content of a victim process running on the Pi. The adversary assumption are as follows:

- The victim's program is executed from the SD card. Access to RAM content is denied for unauthorised users when the program is under execution.
- The adversary is close to the victim device and thus can have physical access. The physical access allows the adversary to swap between victim's and own SD card during the attack.
- The content on victim's SD card is protected (through encryption or other means) and reveals no sensitive information.

Under such assumption of encrypted SD card, it becomes impossible for an adversary to retrieve sensitive information from the SD card with commercial solutions. Under normal operating conditions, if the adversary swaps the victim's SD card with a malicious SD card owned by the adversary, the re-boot process erases the RAM content, thus protecting victim's information from the adversary. This is because the restart of boot momentarily powers down the device thus erasing any content in the RAM instantaneously. As an alternative, adversary can adopt to recover data using cold boot attack as explained in the following.

C. Cold boot attack on IoT device

The recovery of RAM content was shown possible for a laptop if the RAM can be physically separated from the target laptop [1]. The adversary can force the RAM to very low temperatures and transfer it to another laptop where adversary has privileges to dump the victim's RAM content. It is also possible to recover the RAM content without physically removing the RAM, if the adversary can modify the boot process [1]. The latter would not be feasible if boot sequence is locked. However, in several IoT device the RAM is embedded in the CPU or stacked upon the CPU, hindering physical separation. Thus, if the boot sequence in an IoT device clears the RAM upon power-on, cold boot attacks can be prevented in principle, as physical separation of RAM is not possible.

TABLE I
DETAILED SPECIFICATION FOR RASPBERRY PI MODEL B+

Target	**Raspberry Pi model B+** (Pi)
Instruction Set	ARMv6Z (32-bit)
System on Chip	BCM2835 Broadcom with 65nm technology
Central Processing Unit	1 × ARM1176JZF-S 700 MHz
Floating Point Unit	VFPv2; NEON not supported
Memory	512MB (SDRAM) 16 KB (level 1 cache) 128 KB (level 2 cache)

D. Attack Environment

The attack starts with visual identification of the target chip on the board. The chip consist of CPU with RAM stacked over it (refer Table I for specifications). We acquired a off-the shelf low-cost air duster to perform the cold boot attack. The air duster is then sprayed over the chip such that the RAM is frozen as shown in Figure 2, followed by an immediate swapping of victim's SD card with adversary's malicious SD card and rebooting of the chip. The adversary card is programmed to store victim's RAM dump upon boot-up for further off line analysis. Once the victim's RAM dump is recovered, the adversary can restore victim's SD card and boot up to normal operation. The equipment required for the complete attack is under 10 dollars which is comprised of air duster and SD card. We summarize the process of the attack as follows:

1) Adversary aims to recover the secret information from otherwise inaccessible RAM content of the device where victim's program is being executed.
2) The adversary uses air duster to freeze the RAM, switches victim's SD card with a malicious one and recovers the RAM dump of victim's program.
3) The recovered RAM dump is analyzed offline for secret retrieval.

Fig. 2. Demonstration of Raspberry Pi model B+ (Left) frozen by air duster (Right)

The cold boot attack is investigated at the level of L2 cache and SDRAM. We use bare-metal programming and memory is read through USB. The cold boot attack on L2 catch was not successful as it is initialised in the boot sequence as

already mentioned in [8]. The cold boot attack on SDRAM was successful, which also confirm the result of [7]. The details of the attack are given in the next section.

III. Experimental result for Data Recovery From SDRAM

Refer Figure 1 for the boot sequence of the target chip. The SDRAM is enabled by the second stage boot loader and the third stage boot loader (start.elf) is loaded from the SD card onto the SDRAM. An initialization of the SDRAM at this stage (through start.elf) would prevent cold boot attack. Upon investigation, we determined that start.elf does not contain any code for SDRAM initialization unlike L2 cache which was initialised by the boot sequence.

We measure the success of RAM content recovery for various temperatures and model the decay. In the adversary SD card, the kernel.img is modified to read SDRAM via USB communication, enabling cold boot attack. The success of the attack is affected by freezing temperature, and decaying characteristics. In the following, we analyze these technical aspects of attack environment to ensure maximum recovery of information from SDRAM. The experiments were repeated several times to validate consistency of results.

A. Effect of temperature

The impact of temperature on RAM content recovery rate is measured using the famous Mona Lisa image also used by [1]. Figure 3 shows the resulting recoveries. In best conditions, the image can be recovered with a high recovery ratio making recovered image practically infeasible to distinguish from the original image. The pixel wise recovery ratio at a temperature of $-30\,^{\circ}\mathrm{C}$ is reported at 99.718%, where only 3371 bits were recovered with error from the total 1195360 bits. It is potentially possible to increase the recovery rate by forcing temperatures to even lower levels, but the required equipment for that would be costly and thus considered out of scope here. As compared to [1], we observe the decay degradation tends to faster in our setup, which could be related to hardware components.

Moreover, the recovery rate is also affected by the decay time. The decay time is the time elapsed between power-off and power-on of the chip, upon freezing. The recovery ratio for different decay time and temperature is shown in Figure 4. The best and worst recovery rates are 99.718% and 59.7% respectively. It can be observed that the recovery at $0\,^{\circ}\mathrm{C}$ and $-10\,^{\circ}\mathrm{C}$ is very low and comparable to that on normal operating condition of $33.6\,^{\circ}\mathrm{C}$. The recovery ratio increases drastically to 99% at temperatures as low as $-20\,^{\circ}\mathrm{C}$ and decay time of 1 second and drops to 85.5% at 10 second decay time. The attack at $-30\,^{\circ}\mathrm{C}$, allows 99% recovery even for longer decay times.

B. Analyzing Decay Model

Next, to better understand the characteristic of performed cold boot attack, we investigate the decay model. The decay model involves computation of ρ_0 *i.e.* the probability of a bit value 1 flipping to 0 and vice versa ρ_1. To measure ρ_0, we initialize the whole SDRAM to 0xFFFFFFFF, except for firmware and Power-On-Self-Test (POST) spaces. Out of 4169415680 bits (≈497 MB), only 11373 bits are changed to 0 from 1. In the case of ρ_1, 375 bits are only replaced to 1 from 0 after putting whole SDRAM to 0x00000000 (except firmware and POST). The values of ρ_0 and ρ_1 were computed at best attack conditions ($-30\,^{\circ}\mathrm{C}$, short decay time). As summarised in Table II, ρ_0 and ρ_1 are reported around 0.0000027 and 0.00000009, respectively. We further compare the results to [1] in Table III. Note that the temperatures reported in [1] are much lower than obtained in our low-cost setup. The recovery time of 246 seconds does not require the RAM to be frozen for that long. The freezing is required only while the SD cards are replaced which is typically in range of milliseconds.

TABLE II
Decay Model and Elapsed Time for recovering whole RAM

Decay Model	ρ_0 $(1 \rightarrow 0)$	ρ_1 $(0 \rightarrow 1)$
Magnitude (at $-30\,^{\circ}\mathrm{C}$)	0.0000027 ($\approx$ 11373/4169415680)	0.00000009 ($\approx$ 375/1)
Elapsed Time	246 seconds for 512 MB	

TABLE III
Comparison between our target and previous targets [1]

Device	**Temperature**	**Error Rates (%)**
Device A [1]	$-50\,^{\circ}\mathrm{C}$	0.000095
Device B [1]	$-50\,^{\circ}\mathrm{C}$	0.000036
Device C [1]	$-42\,^{\circ}\mathrm{C}$	0.00144
Device D [1]	$-50\,^{\circ}\mathrm{C}$	0.18
Our Target	$-30\,^{\circ}\mathrm{C}$	0.00036

IV. Attacking Encrypted Disks

One of the prime target of cold boot attack is data encryption. This is because, although much advancement is done in the domain of encryption and security, it is still very difficult to perform computations over encrypted data. Thus, while sensitive data can be kept encrypted when not used, if any computation is required over the data, it must be loaded to SDRAM, decrypted, computed upon, re-encrypted and stored back. To enable this encryption and decryption in the SDRAM, the secret key must be loaded into the SDRAM as well. A targeted cold boot attack can be performed when the secret key is in the RAM to dump the RAM content and recover the secret key, thus bypassing all security measures. Modern processors propose special hardware for protecting the key, but this is not the scenario in low cost IoT devices like the Pi.

To demonstrate the practicality of cold boot attack for recovering secret key used to encrypt disk, we choose dm-crypt as our target, which contains LUKS (Linux Unified Key Setup) format, is usually used as one of encryption solutions in Linux. Attack on dm-crypt was shown in [1] for an old version of Linux on standard PC but never on a IoT device like Pi with stacked memory. Moreover, some recent applications such as LibreCrypt and FreeOTFE offer the LUKS-compatible disk

Fig. 3. A MonaLisa image is stored to SDRAM. After powering off for 5 seconds (leftmost), 30 seconds, 60 seconds, 5 minutes (rightmost) at −30 °C, the image is recovered via cold boot attack. The leftmost image is indistinguishable from the original.

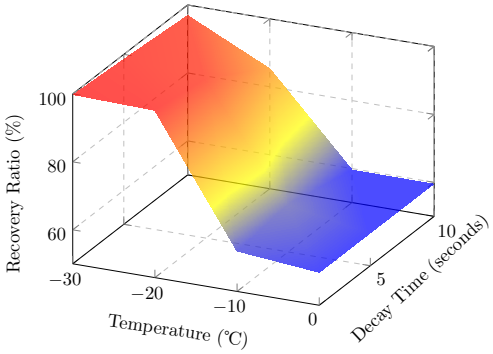

Fig. 4. Impact of decay time and temperature on the recovery ratio of cold boot attack

encryption for Windows OS. We target dm-crypt encryption solution with linux kernel version 4.19 and default cipher AES-XTS-512 with 512-bit master key. The attack scenario is as below.

1) (Victim) Encrypts a disk (like USB drive or similar) based on dm-crypt with default setting.
2) (Adversary Assumption) The encrypted disk is in use.
3) (Cold Boot Attack) After freezing the RAM and changing the adversary's SD card, an adversary steals the encrypted drive content and RAM data. With the recovered secret key from RAM data, the drive content can be independently decrypted.

The experimental setting and recovery process are as reported in the last sections. Even though we use recent kernel version and advanced default cipher as compared to those reported a decade ago [1], we can recover the 512-bit master key[1]. This reinstates that even state of the art devices are vulnerable to long known attacks on different platforms.

The two 256-bits key, used in two different instances of AES-256, are stored in different sections of the RAM which must be recovered. One of two different AES-256 key is represented in Figure 5.

[1]Method for recovery of secret key from RAM content can follow aeskeyfind [1]

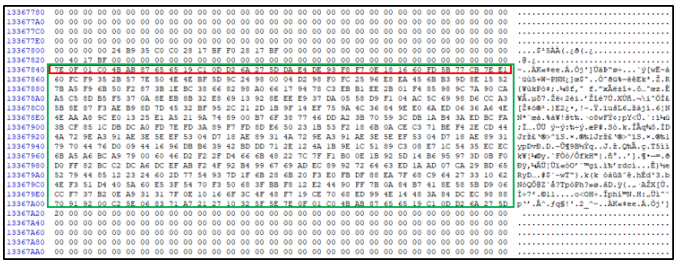

Fig. 5. Dumped memory for one of two different AES-256 key

The red box indicates the master key and the green box is round keys for AES-256. This key can be identified exploiting the property that master key and round keys are located in consecutive memory locations. This was also exploited in aeskeyfind program (suggested in [1]). As seen in Figure 5, the round key is stored at consecutively memory address can be identified.

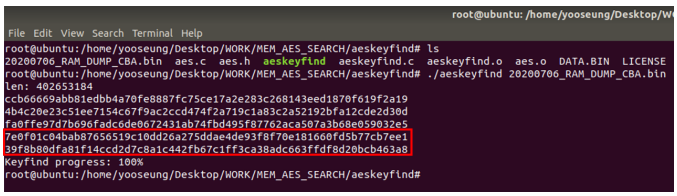

Fig. 6. aeskeyfind program result for dumped memory

After recovering the 512-bit master key, we create the binary file to open the encrypted USB using the following command: cryptsetup –master-key-file [Key File Name] luksOpen [Device Location] [Driver Name]
by providing key file location, encrypted driver identifiers and associated drivers.

The command cryptosetup luksClose in the latest versions of the dm-crypt ensures erasure of key when operations with the encrypted drive are terminated. However, as in our experiments it can still be attacked during operations with the drive.

V. Mitigation & Conclusion

A. Mitigation

1) IoT Device: IoT devices with rather limited security features can be a victim of cold boot attacks. This is often motivated by adoption of old memory technologies driven by the low-cost requirements. However, a systematic and secure boot process can be put in place to overcome such vulnerabilities, ensuring memory initialization and prevent unauthorised modification of boot sequence/firmware. The problem can grow further with emerging memory technologies based on non-volatile magnetic memories [9]. The non-volatility will enable easy RAM dump recovery as RAM data would be retained despite power-off. Thus, secure boot should trigger proper erasure and protection of sensitive RAM content.

2) Encrypted Disk Solutions: Even though the device is vulnerable to cold boot attacks, there are some solutions to protect the disk, utilizing the safe encryption solutions. Some software solutions such as TRESOR [10] and ARMORED [11] can be employed to acquire the resistance against cold boot attacks since only processor registers are used as storage of master key at a performance penalty of $1.5 - 4.5\times$.

B. Final Remarks

We demonstrate vulnerability of IoT device with stacked RAM against extremely low-cost cold boot attacks, allowing recovery of encryption passwords. It highlights the urgent need for strong but resource friendly and effective secure boot solutions to protect billions of devices to be deployed in the future.

Acknowledgment

This research is supported in parts by the National Research Foundation, Singapore, under its National Cybersecurity Research & Development Programme / Cyber-Hardware Forensic & Assurance Evaluation R&D Programme (Award: NRF2018NCR-NCR009-0001).

References

[1] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten, "Lest we remember: cold-boot attacks on encryption keys," *Communications of the ACM*, vol. 52, no. 5, pp. 91–98, 2009.

[2] M. Tilo, S. Michael, and F. C. Freiling, "Frost: forensic recovery of scrambled telephones," in *Proceedings of the International Conference on Applied Cryptography and Network Security*, 2014, pp. 373–388.

[3] J. Bauer, M. Gruhn, and F. C. Freiling, "Lest we forget: Cold-boot attacks on scrambled ddr3 memory," *Digital Investigation*, vol. 16, pp. S65–S74, 2016.

[4] S. F. Yitbarek, M. T. Aga, R. Das, and T. Austin, "Cold boot attacks are still hot: Security analysis of memory scramblers in modern processors," in *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2017, pp. 313–324.

[5] O. Segerdahl and P. Saarinen, "An ice-cold boot to break bitlocker." in *BlueHat Security Conference*, 2018.

[6] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis *et al.*, "Understanding the mirai botnet," in *26th {USENIX} security symposium ({USENIX} Security 17)*, 2017, pp. 1093–1110.

[7] Y.-S. Won, J.-Y. Park, D.-G. Han, and S. Bhasin, "Practical cold boot attack on iot device-case study on raspberry pi," in *2020 IEEE International Symposium on the Physical and Failure Analysis of Integrated Circuits (IPFA)*. IEEE, 2020, pp. 1–4.

[8] "Broadcom bcm2835 arm peripherals." [Online]. Available: https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2835/BCM2835-ARM-Peripherals.pdf

[9] N. Nishimura, T. Hirai, A. Koganei, T. Ikeda, K. Okano, Y. Sekiguchi, and Y. Osada, "Magnetic tunnel junction device with perpendicular magnetization films for high-density magnetic random access memory," *Journal of applied physics*, vol. 91, no. 8, pp. 5246–5249, 2002.

[10] T. Müller, F. C. Freiling, and A. Dewald, "Tresor runs encryption securely outside ram." in *USENIX Security Symposium*, vol. 17, 2011.

[11] J. Götzfried and T. Müller, "Armored: Cpu-bound encryption for android-driven arm devices," in *2013 International Conference on Availability, Reliability and Security*. IEEE, 2013, pp. 161–168.