Thermal Covert Channels Leveraging Package-on-Package DRAM

Conference Paper · August 2019

DOI: 10.1109/TrustCom/BigDataSE.2019.00050

5 authors, including:

Shuai Chen
Southeast University (China)
9 PUBLICATIONS **38** CITATIONS
SEE PROFILE

Wenjie Xiong
Yale University
26 PUBLICATIONS **180** CITATIONS
SEE PROFILE

Yehan Xu
Southeast University (China)
3 PUBLICATIONS **4** CITATIONS
SEE PROFILE

Jakub Szefer
Yale University
66 PUBLICATIONS **1,091** CITATIONS
SEE PROFILE

Thermal Covert Channels Leveraging Package-On-Package DRAM

Shuai Chen*, Wenjie Xiong†, Yehan Xu*, Bing Li* and Jakub Szefer†
*Southeast University, Nanjing, China
{chenshuai_ic,220174472, bernie_seu}@seu.edu.cn
†Yale University, New Haven, CT, USA
{wenjie.xiong, jakub.szefer}@yale.edu

***Abstract*—Package-on-Package (PoP) is an intergraded circuit packaging technique where multiple separate packages are mounted vertically one on top of the other, allowing for more compact system design and reduction in the distance between modules. However, this can introduce security vulnerabilities. In particular, this work shows that due to the close physical proximity of a System-on-a-Chip (SoC) package and a PoP DRAM that is on top of it, a thermal covert channel exists between the SoC and the PoP DRAM. The thermal covert channel can allow multiple cores in the SoC to communicate by modulating the temperature of the PoP DRAM. Especially, it is possible for one core in the SoC to generate heat patterns, which encode data that is to be transmitted, and another core to observe the transmitted pattern by measuring the decay rate of DRAM cells. Further, a covert channel can be established between a remote user and an SoC core by modulating the PoP DRAM temperature that then affects errors in PoP DRAM PUF fingerprints that are sent to the remote user for device authentication. Following these ideas, this paper introduces two new covert channels: first leveraging the observation of the number of bit flips in the PoP DRAM directly, and second through observation of bit flips induced in PoP DRAM PUF fingerprints. This paper also evaluates the DRAM cell decay rates in PoP DRAM, when heated by execution of different instructions on the processor cores in the SoC underneath the DRAM module. The evaluation was performed on three Raspberry Pi B+ boards, which have PoP DRAM. The evaluation was done both in office environment and in a thermal chamber. To mitigate the new channels, a set of defenses is discussed.**

1. Introduction

A covert channel is an attack technique used to leak data between different programs via a channel not originally designed for information communication [1]. The covert channel is "not intended" for communication, which makes exfiltration of information through a covert channel difficult to detect and prevent as the entity trying to find the leaks has to first guess what is the communication channel, before being able to observe and prevent it. Over the last two decades, a variety of covert channels have been shown in computer systems. The channels can be based on observation of execution time [2], [3], or on observation of physical emanation, such as heat [4] or electromagnetic (EM) field [5], for example. This work focuses on thermal channels, and shows how the heat, or temperature, can be measured without special measurement equipment or physical access in commodity SoC devices. Especially, this work focuses on an SoC DRAM Package-on-Package (PoP) configuration [6] where two chips (SoC and DRAM) are stacked on top of each other. The heat transfers between the two can be observed by measuring decay rate of DRAM cells, which is the basis for this work.

Previously, heat-based or thermal covert channels have been demonstrated in data centers [7], or in multicore processors [8], but these require dedicated thermal sensors unlike our work. Also, it has been demonstrated that an attacker can use thermal covert channel for data transmission between electrically and physically isolated circuits in FPGAs [4], [9], but these require use of FPGAs where custom circuits can be implement to measure the temperature. Unlike the prior, we demonstrate covert channels in SoC configuration with PoP DRAM, without any dedicated thermal sensor nor need for reconfigurable FPGAs.

As it has been previously show [10], [11], [12], the decay rates of DRAM cells are very sensitive to temperature changes. In this work, we show that the temperature changes due to SoC operation affect the decay rate of cells in PoP DRAM, e.g., higher temperature results in faster decay. When reading DRAM with refresh disabled, the decay rate can be observed by counting the number of bit flips that have occurred in a DRAM region. The DRAM decay rate then contains some information about the amount of heat generated depending on the program being executed on the SoC. The system with an SoC and PoP DRAM used in this work is shown in Figure 1.

This can be used as a covert channel: one processor core can be the data sender, and depending on whether zeros or ones are to be transmitted it can execute some specific operation in a certain order to heat up DRAM chip for predefined periods of time. For the data receiver, the data can be decoded by evaluating the decay rate of DRAM cells in the DRAM atop of the SoC. The decoding can be done locally in the first covert channel, or remotely by evaluating errors in PoP DRAM Physically Uncloneable Function (PUF) fingerprints in the second covert channel.

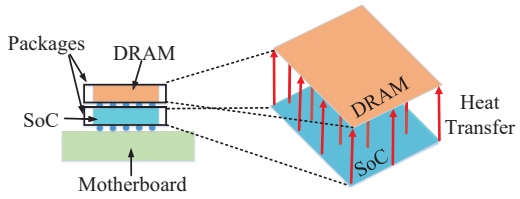


Figure 1. The architecture of an SoC with a PoP DRAM. The whole DRAM chip can be heated up by performing different operations of the SoC.

1.1. Contributions

Our key contributions are as follows:

- Design of two novel covert channels leveraging Package-on-Package DRAM:
  - By measuring the number of bit flips of the PoP DRAM while the DRAM refresh is disabled for a specified amount of time, the first channel can be used to enable cores located in separate areas of the SoC to transmit data between each other even if they are separated by logical isolation.
  - By measuring the errors in PoP DRAM PUF fingerprints, the second channel can be used to reveal secret data to external systems that receive the fingerprint.
- Evaluation of the PoP DRAM decay rates under different ambient temperatures with different DRAM decay times, and while different programs execute on processor cores in the SoC beneath the DRAM to find most suitable configuration for the covert channels.
- Discussion of a set of defenses that can be used to mitigate the new PoP covert channel vulnerabilities.

2. Background

This section provides a brief background on DRAMs, DRAM cell decay, application of DRAM cell decay such as PUFs, and the Package-on-Package packaging technique.

2.1. DRAM Decay and PUFs

DRAM is pervasively used in existing embedded systems. As shown in Figure 2, each DRAM cell consists of a transistor and a capacitor, which is one bit of data. The cells are arranged in a 2-dimensional array, where each row of the array is connected to a word line. In each column, cells are linked by a bit line. Amplifiers and equalizers are used to help read the data and convert it to voltage levels corresponding to logical ones or zeroes. In DRAM, the capacitors leak charge over time (some of the leakage paths are shown in Figure 2), which may cause the data to flip if the capacitor goes from charged to discharged state. Therefore, DRAM chips usually have a periodic self-refresh controlled by the memory controller to ensure the cells are refreshed before the bits can flip.

The rate at which the charge on each capacitor leaks is dependent on manufacturing variations, temperature

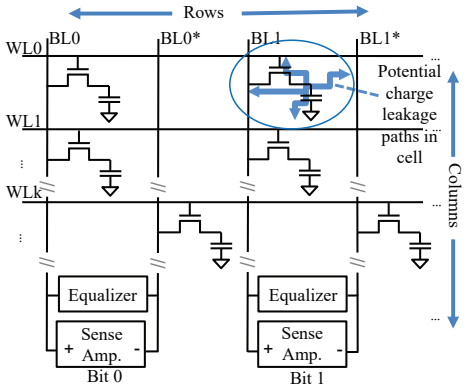


Figure 2. Layout of a typical DRAM array, arrows indicate the leakage paths that may lead to DRAM decay.

changes, and voltage changes. Especially, some cells leak faster than others. After a certain decay time, if enough charge has leaked, the weak cells will flip their value if they were previously in a charged state. But for the other strong cells, the contents will stay stable. Due to the manufacturing variation, DRAMs have been shown to be a good candidate for use as PUFs [10], [13], [14].

2.2. PoP DRAM

In some devices (e.g., mobile phones and IoT devices), a special integrated circuit packaging method called Package-on-Package (PoP) is used to improve the component density. For example, as shown in Figure 1, the DRAM chip can be mounted on top of the SoC chip. Usually, the logic package is on the bottom because it needs more connections to the PCB board. Compared with the traditional isolated chip packaging, PoP yields better PCB space savings and shorter electrical connections between SoC and the memory chip with better electrical performance.

However, this packaging method may become the target of a security attack. Operation of the cores in the SoC affects the DRAM on top through thermal changes, and this can be used to leak information. This is a new threat, compared to other systems designs where DRAM and SoC are separate chips in different parts of PCB and one cannot easily affect temperature of the other.

3. Covert Channel using Bit Flips in DRAM

Figure 3 describes the first covert channel presented in this paper. The SoC chip on the RaspberryPi B+s contains a CPU and a GPU. To demonstrate that the covert channel can send information between distinct cores in the SoC, the GPU is used by the sender (Alice), while the CPU is used by the receiver (Bob).

The sender, Alice, is a program running on GPU and the receiver, Bob, is a program running on the CPU. Both are isolated logically, and cannot sent data undetected over other channels. Especially, the programs running on the

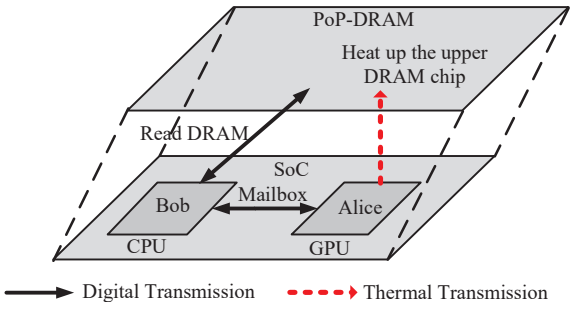


Figure 3. Block diagram of the covert channel based on the number of raw bit flips in PoP DRAM.

two cores are only able to communicate via the mailboxes. As the mailbox accesses and timing can be monitored, we assume covert channels through mailboxes are not possible. However, Alice can try to transmit data by modulating the heating of the DRAM chip (the dotted arrow in Figure 3), thus bypassing the system's protections.

The transmission process consists of two basic parts. In the data encoding process, Alice encodes the data into a sequence of functions to be executed on the GPU. Depending on whether a zero or a one is to be transmitted, the GPU performs different functions for predefined periods of time. Especially, the GPU becomes a circuit that generate heat according to the function that is executed. To heat up the GPU, a loop of function calls is executed. Depending on the functions that are executed, the GPU heats up in a different manner. This heat transfers to the PoP DRAM.

Once the PoP DRAM is heated, the thermal transmission can be detected by measuring the decay rate of the PoP DRAM. Bob can initialize the DRAM with known data, disable the refresh of the DRAM module, and allow the memory to decay. Then, he can read out DRAM using normal memory accesses (the solid arrow in Figure 3). After reading the DRAM, Bob can compute the number of bit flips. From the number of bit flips, and knowing type of DRAM chip used, Bob can compute how much the DRAM was heated up by Alice. Based on a threshold, Bob can decide if a logical 1 or logical 0 was transmitted. To transfer more bits, the process can be repeated for each bit.

Due to the fast heat spreading in SoC and DRAM, it is not possible, based on our evaluation, to transfer more than 1 bit at a time. I.e., is is not possible to heat different regions of the DRAM in a different manner. However, for the usual case of trying to leak secrets such as cryptographic keys, it is only necessary to repeat the process a small number of times, typically corresponding to the secret key size.

Details of a covert channel transmission are shown in Algorithm 1, where Alice tries to exfiltrate n-bit secret data to Bob via the PoP DRAM covert channel. For $i \in 1, \cdots, n$, if $k_i = 0$, Alice executes function A, else she executes function B to transfer a 0 or a 1, respectively. Function A and function B are different functions, which heat up the DRAM in a distinguishable manner. The t_{send} should be strictly bigger than the decay time T discussed below,

Algorithm 1: Covert channel protocol using raw bit flips in PoP DRAM.

k: n-bit message to be sent by Alice
k': n-bit message to be received by Bob
t_{send}: predefined time for transmission of 1 bit
T: decay time
ads, ade: start and end address of DRAM region used by Bob
r: number of bit flips observed by Bob
ξ: threshold for differentiating logical 1 and 0

The sender Alice:

```
for i = 0; i < n; i = i + 1 do
    for an amount time t_send do
        if k_i = 0 then
            do function A in a loop;
        else
            do function B in a loop;
        end
    end
end
```

The receiver Bob:

```
for i = 0; i < n; i = i + 1 do
    Initiate DRAM from ads to ade to a known value,
      such as all 0s;
    Disable the DRAM refresh;
    for an amount time T do
        nothing;
    end
    Enable the DRAM refresh;
    Read DRAM from ads to ade and count the bit flips
      r;
    if r > ξ then
        k'_i = 1;
    else
        k'_i = 0
    end
end
```

so the receiver has time to measure the DRAM decay, and deduce the temperature.

In the CPU, at the same time, Bob initializes the DRAM region from start address ads to end address ade to all zeros[1]. Then, the evaluation of DRAM decay is started by disabling the DRAM refresh[2]. After a decay time T has elapsed, the decay process ends and the DRAM refresh can be re-enabled. Finally, Bob decodes the data according to the number of bit flips r based on the threshold ξ: if $r > \xi$, $k'_i = 1$, else $k'_i = 0$. Then Bob waits for the t_{send} to end, and repeats his operations to read the next secret bit.

In this channel, Bob's operation should be performed in kernel or have the root privileges so Bob can control DRAM refresh operations. If Bob is not able to directly control DRAM refresh, he and Alice may still be able to

1. Other patterns are possible, but initial zeros have worked well in our experiments, as they set at least half of DRAM cells into charged state.

2. If there is useful data in other parts of DRAM not used for the measurements, the other DRAM regions should be manually refreshed to prevent that data from decaying, and to maintain the operation of the system. This can be done without any hardware changes, following prior work [11].

try to create the covert channel by leveraging power saving modes of the SoC where the SoC turns off the DRAM to save power. If Bob can trigger or predict when low power mode happens, and for how long, he can write the zeros before DRAM is disabled and read data from DRAM when it is re-enabled to observe the decay rate.

3.1. Choice of Encoding Functions

Different choices for *function A* and *function B* are possible. In the basic case, Alice could modulate the GPU between some operation and no operation. But this would allow for a user or a system administrator to observe unusual GPU patterns. For example, through power measurements they can learn GPU is behaving in a special pattern of operation and no operation, and thus detect the existence of the covert channel.

Our work focuses on a more stealthy channel. For example, a loop of addition operations vs. a loop of division operations can be used to send either 1 or 0. Specifically, each of these operations takes different amount of time to execute, loop of additions is quicker, and causes GPU to do overall more work per unit time, and heat up more. Meanwhile divisions are slow, and cause GPU to do overall less work per unit time, and heat up less. Our evaluation in Section 5.4 shows there is 5% difference between heating due to the loops of two types. Thus, Alice and Bob can send information while always keeping the GPU busy, and making it harder to spot that some transmission is going on.

3.2. Run-Time Detection of Decay Threshold

If the threshold ξ for decoding is not known prior to the transmission previously, some fixed known data, e.g., $0101010101\cdots$, can be transformed from Alice to Bob. Knowing this data is being transmitted, Bob can compute the threshold of ξ by averaging the number of bit flips of the known data. Thus, before receiving data, Bob has to synchronize with Alice, and then pre-compute the threshold. This is the setup overhead for setting up covert channel for transmission each time. Also, if the temperature is varying over time, Bob can receive all the data bits, then compute the envelope [15] and decide which bits are logic 1 and logic 0 depending if the decay rate is closer to the upper or lower envelope. This does not require pre-sending a fixed data pattern. Details are given in Section 5.

4. Covert Channel using Bit Flips in DRAM PUF Fingerprints

DRAM PUFs can be used as the fingerprints of devices for authentication. Usually, the PUF-based authentication has two steps: an enrollment phase in a secure environment, and an authentication phase when the device is used in the field. Most PUFs exhibit an unreliability problem due to aging and the inherent sensitivity to the environmental conditions, such as temperature. As remedy to the reliability issue, error-correction algorithms [16] or helper data

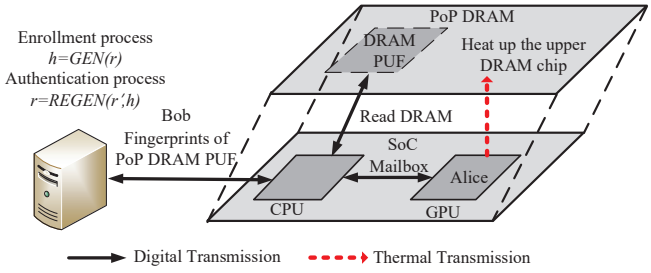


Figure 4. Block diagram of the covert channel based on the fingerprints of the PoP DRAM PUF.

algorithms [17] are used in practice. A helper data algorithm generates and stores the helper data in the enrollment phase. The generated helper data are used then for error correction in the authentication phase. Our second covert channel design exploits errors that can be detected in the authentication process, to stealthily send data to remove receiver under the guise of sending the device's fingerprint.

Figure 4 shows the second covert channel. In the enrollment process, Bob generates helper data $h = GEN(r)$, where GEN is the generation process of helper data algorithm and r are the raw DRAM PUF fingerprint measurements. As with any helper data, h can be stored publicly as it should not leak any information about the fingerprint. In the authentication process, Bob queries the DRAM PUF from the target device and gets a measurement of the DRAM PUF fingerprint *r'*. Using the helper data algorithm, Bob can regenerate the fingerprints $r = REGEN(r', h)$, where $REGEN$ is the error correction process of helper data algorithm, if the Hamming Distance of *r* and *r'* is smaller than the error-correcting capability of the help data algorithm.

For the second covert channel, Alice again executes code on the GPU to heat the DRAM and transmit some secret data to Bob, but now Bob is a remote user who has only access to the DRAM PUF fingerprint authentication data sent form the device where Alice is executing her code. Depending on whether 0 or 1 are to be transmitted, Alice performs different operations on the GPU for a predefined period of time, t_{send}. She can heat up the SoC a lot (e.g., using loops of additions) or heat it up less (e.g., using loops of divisions). Each one will affect DRAM decay in a different manner, and thus change the DRAM PUF fingerprint in a different way.

Then, during the authentication phase, Bob receives the fingerprint data. But the data has been affected by this time due to the decay rate change of the DRAM because of Alice's operations on the GPU. With the fingerprint data, Bob can perform both the authentication (apply error correction to recover the secret) and also compute the magnitude of the error introduced due to heating. The magnitude will reveal whether Alice was transmitting a 1 or a 0. This will always work if the changes in the decay rate of DRAM due to Alice's actions cause less error than can be handled by the error-correction function, otherwise the fingerprint will be affected as well and the authentication will fail.

5. Implementation and Evaluation

The covert channels were evaluated on commodity off-the-shelf Raspberry Pi B+ devices, without any hardware modifications. Evaluation considers both ideal thermal conditions (in a thermal chamber) and more realistic varying thermal conditions (in an office room).

5.1. Implementation in Raspberry Pi B+

We implemented and tested our covert channels on three Raspberry Pi B+ development boards, labeled *Pi1*, *Pi2*, and *Pi3*. Raspberry Pi B+ uses a Broadcom BCM 2835 System-on-Chip (SoC) module that includes multiple cores. The CPU is a 700MHz ARM11 76JZF-S processor, while the GPU is a VideoCore IV. The Raspberry Pi B+ also includes 512MB of DDR2 memory in the Package-on-Package configuration atop the SoC.

In order to control the refresh rate, the firmware of the Raspberry Pi B+ needs to be modified. We modified the open source Raspberry Pi B+ firmware [18] in order to allow for controlling the DRAM refresh from software. On the Raspberry Pi B+, the refresh is controlled at the granularity of the whole DRAM chip. It is not possible to control the refresh of only part of the DRAM module. Thus, the firmware must selectively refresh the memory range which is used by the other applications. Similar to existing work [10], [11], a software approach is used. A loop over all memory address that need to be refreshed is used. In the loop, reads are issued to the first word in every DRAM row of the memory locations that need refresh. Recall, that each DRAM read automatically refreshes the DRAM cells. If a different SoC were used with multiple separate DRAMs, or with features allowing control of refresh at a finer granularity, the software loop would not be needed.

For both channels, Alice's application runs on the GPU. For the covert channel using PoP DRAM decay, Bob's application runs on the CPU. For the covert channel using PoP DRAM PUF fingerprint, a local application runs on the CPU to collect the PUF measurements, and Bob's application runs on the workstation.

5.2. Experimental Setup

Figure 5 presents schematic of the experimental setup used to verify the feasibility and reliability of the covert channels. This setup includes the following equipments. A thermal chamber with an Ethernet interface. Inside the thermal chamber, three Raspberry Pi B+ boards are placed for experiments under controlled temperature. The Raspberry Pi B+ boards communicate with the workstation via the serial ports. The workstation further runs Python scripts for controlling the thermal chamber and the Power Distribution Unit (PDU). The server is used to allow for remote control of the PDU to power on and off the boards for testing, server accepts commands over SSH and controls the PDU via a serial port.

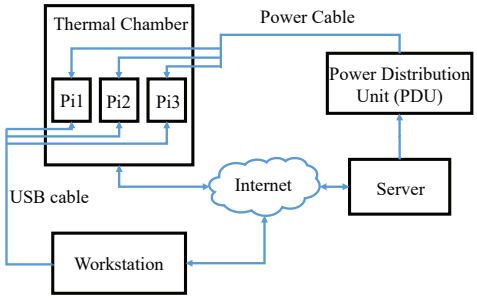


Figure 5. The experiment setup. The workstation can control the remote thermal chamber and the PDU which is attached to the server, and gather data for the covert channels using PoP DRAM bit flips and PoP DRAM PUF fingerprints.

We performed measurements using three different Raspberry Pi B+ boards in the thermal chamber under different temperatures and decay times. The thermal chamber can be controlled remotely over the Ethernet, e.g., setting or reading the temperature. The Raspberry Pi B+ boards were powered by the PDU. The workstation running the python scripts controlled the whole experimental setup. The measurements were gathered following the automated process:

1) The workstation first sets the temperature of the thermal chamber and starts to monitor the temperature inside.
2) When the temperature requirements are reached, the workstation powers on the Raspberry Pi B+ boards by turning on ports on the PDU.
3) The workstation sets the current test's decay time (used by Bob's application) of the DRAM in Raspberry Pi B+ boards using the serial port.
4) Once the decay time expires, measurements of DRAM decay are collected:
   - For the evaluation of the covert channel based on the bit flips in PoP DRAM, Bob's application running on the CPU reads the (decayed) DRAM data and sends out the results to the workstation over the serial port to inform the controller script of the observed number of bit flips (and thus the secret data bit for the current experiment).
   - For the covert channel based on the PoP DRAM PUF fingerprints, application on the CPU reads the fingerprint from the DRAM PUF, and sends it out via serial port to the workstation. In the workstation, Bob's application decodes and error corrects the data based on the helper data system. It also recovers the secret bit for the current experiment from the data based on the amount of error observed.
5) Restart from step 1 for next set of parameters.

For the experiments in room setting, the thermal chamber is not used, and the Raspberry Pi B+ boards are placed on a desk in an office environment.

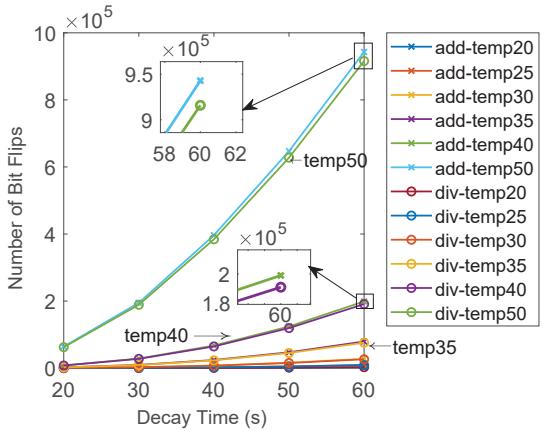


Figure 6. Evaluation of number of bit flips when different functions are running in a loop on the GPU under different decay times and at different ambient temperatures (as controlled by the thermal chamber). The data are shown for *Pi1* (the first Raspberry Pi B+ board), since all three boards show similar behavior. The difference between the two functions (addition and division) is reliably about 5%.

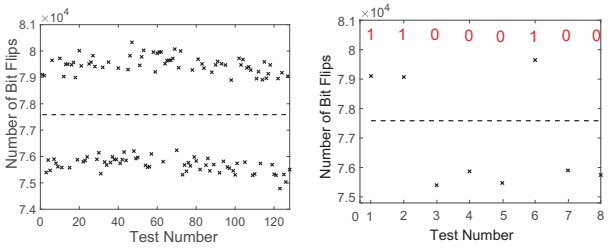
5.3. Changes in PoP DRAM Bit Flips due to Operations on the SoC

We measured the decay rate of DRAM cells on three instances of Raspberry Pi B+ boards to understand how they are affected by the different operations on the SoC. Given the large amount of memory present, we measured multiple DRAM rows, totaling 16MB on each device. The rows were spread across the whole DRAM to get average decay rate across whole DRAM chip. As shown in Figure 6, each DRAM was measured at seven temperatures $t = 20°C, 25°C, 30°C, 35°C, 40°C, 50°C$, with five decay times $T = 20s, 30s, 40s, 50s, 60s$, and two functions (a loop of addition operations and a loop of division operations, both performed on the GPU) with 20 measurements each.

As shown in Figure 6, although the temperature and the decay time affect the number of bit flips significantly, they do not change the relative bit flips due to the two functions performed on the GPU. We observed that the number of bit flips for addition is always larger (about 5%) than division. This can be explained because addition is quick and can be executed more times than division within a certain decay time. Therefore, the SoC chip has more power consumption and thermal emanation. Furthermore, we observed that the whole chip was heated up with uniform regularity. This enabled us to detect the behavior of SoC by evaluating the decay feature of any part of DRAM in the whole DRAM chip. This shows that executing operations on the GPU can heat up the DRAM in a predictable manner.

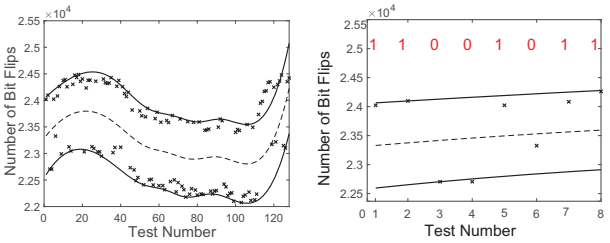
5.4. Environmental Effects on PoP DRAM Bit Flips

The PoP covert channels can be used to transmit secret data to other cores on same SoC, by evaluating the bit flips of PoP DRAM, as well as to external devices by counting the number of errors in PoP DRAM PUFs fingerprints.



(a) 128-bit data transmission (b) Detail of the first 8 bits sent

Figure 7. The covert channel based on bit flips at 35°C, with 60s decay time.



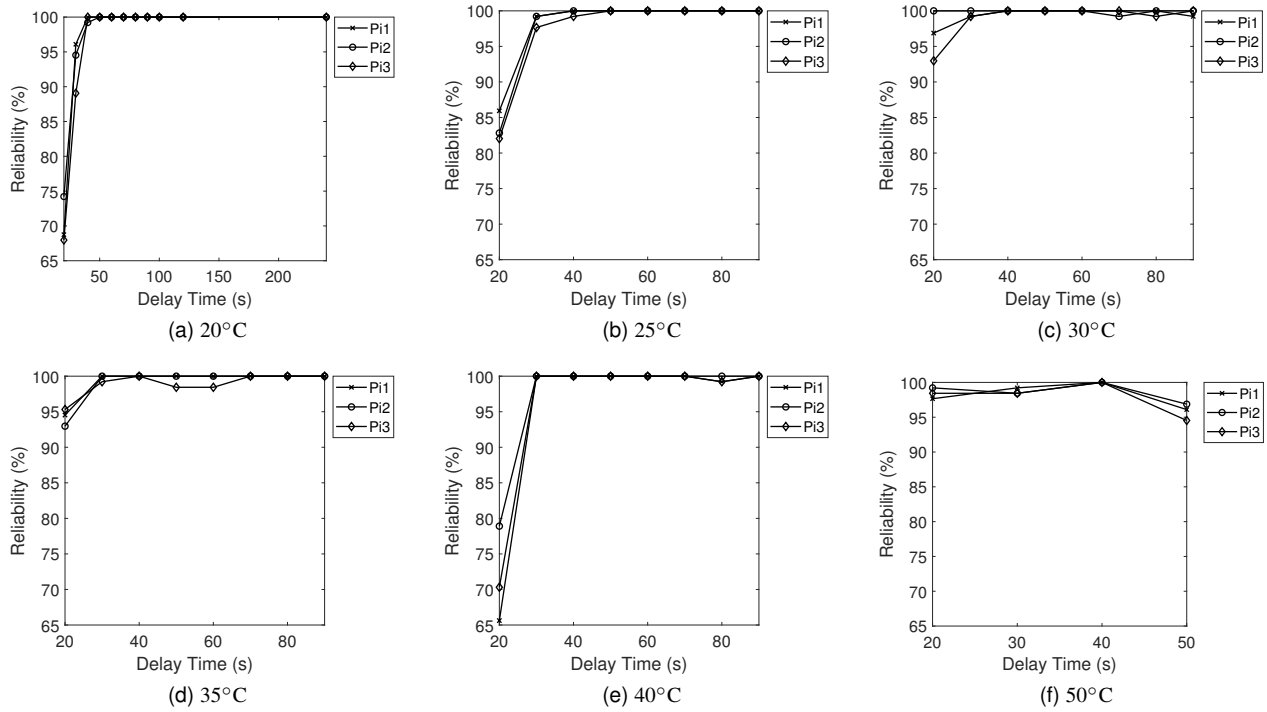
(a) 128-bit data transmission (b) Detail of the first 8 bits sent

Figure 8. The covert channel on bit flips at room temperature, with 60s decay time.

Although the scenario of internal and external channels are different, they are all based on the decay behavior of cells in PoP DRAM. Therefore, by evaluating the PoP DRAM under different test conditions, we can verify the feasibilities of our covert channels.

In Figure 7, we test *Pi1* at constant $t = 35°C$ controlled by the thermal chamber, with $T = 60s$ decay time. Here, the GPU core runs different functions constantly in a loop to transmit 128-bit data, by using different heating levels to transmit a 1 and a 0. Each point in Figure 7(a) represents 1 bit of data, the value is evaluated by examining the number of bit flips. To establish a threshold, a preamble of 14-bit data ("10101010101010") is first transmitted. The dotted line is the average of the number of bit flips for the 14-bit data, the data is not shown in the figure. The points above the dotted line are judged to be a logical 1, while the others are a logical 0. The gap between 1s and 0s is stable under constant temperature. Figure 7 (b) shows the number of bit flips in DRAM of each of the first 8 bits transmitted.

However, as shown in Figure 8, at an ambient room temperature in an office setting, the absolute value of bit flips changes under the influence of temperature changes. The average value of the 14-bit data preamble cannot be used as the threshold for the covert channel in this setting. Fortunately, the relative value of bit flips corresponding to logical 1s and 0s remain stable, because the chip temperature does not experience abrupt thermal changes. Therefore, the envelope [15] can help us separate logical 1s and logical 0s more accurately. Specifically, the data has to be first gathered for the whole message (while for stable temperature each bit can be decoded right away when it is received and can



(a) 20°C (b) 25°C (c) 30°C

(d) 35°C (e) 40°C (f) 50°C

Figure 9. The reliability of covert channel at different temperatures with same decay time: 60s.

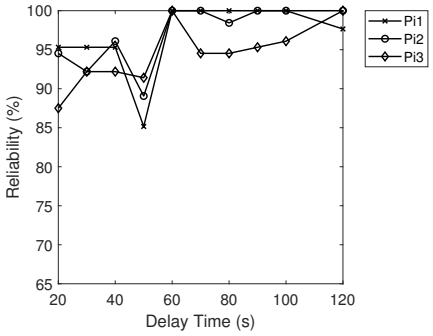


Figure 10. The covert channel at room temperature, 60s

be judged to be above or below the threshold). Once the data are gathered, the envelope (shown by the solid line) is computed. Then, the average of the upper and lower envelope can be computed at each time point (shown by the dotted line). Finally, bits above the varying average line are computed to be logical 1s and below to be logical 0s. Figure 8 (b) shows the number of bit flips in DRAM of each of the first 8 bits transmitted.

5.5. Reliability Evaluation

To understand the reliability of the covert channels, we evaluated the covert channels reliability using both the thermal chamber and at ambient room temperature. The reliability evaluation was done using Formula 1 shown below, where HD is the Hamming Distance.

$$reliability = \frac{HD(k, k')}{n} \times 100\% \quad (1)$$

Figure 9 shows the reliability of our covert channel with three Raspberry Pi B+ boards operating at $t = 20°C, 25°C, 30°C, 35°C, 40°C, 50°C$ and decay time T from $20s$ to $240s$. For the temperature $t = 20°C, 25°C, 30°C$, it is shown that the reliability of our covert channels is getting better as the temperature increases. As shown in Figure 6, when the chip is working below $30°C$, the relative bit flips between two functions are too small to recognize. As the temperature increases, the differences of the decay rates gradually become larger. From 30°C to 40°C, the reliability of covert channel fluctuates, especially for the *Pi3*. The chip is now working under the environment temperatures that are close to the chip's internal temperature. And any excess heat will not be absorbed by the environment, therefore, covert channel presents the best test results under these experimental conditions. As the temperature gets even higher, e.g., 50°C, the reliability gets worse. As shown in Figure 6, the slope of each line increases as the temperature increase. That means even little environment temperature fluctuations will lead to higher decay rate changes, which may be bigger than the changes due to operations executed on the SoC. Note, even in thermal chamber, the temperature is not fully stable due to internal heaters and fans that turn on and off to circulate air and to stabilize the temperature.

The test results at room temperature are shown in Figure 10, despite ambient thermal noise, the reliability of

covert channels can reach 95%, with the decay time greater than 60s. To further increase the reliability of the channel, Alice and Bob can use extra error correction to compensate for the errors (at cost of having to send extra data bits for the error correction algorithm).

6. Defense Strategies

A natural defense to prevent our covert channels would be to limit the thermal conduction between the SoC and the DRAM. E.g., add more vertical space between the PoP chips or add more thermal insulation between the PoP chips. This would create higher thermal resistance and the heat would not conduct from the SoC to the DRAM as easily.

Another defense strategy would be to heat and keep the DRAM a certain temperature. For example, circuits such as Ring Oscillators can be used as heaters [19] and could be added to the DRAM chip. As long as SoC could not heat the DRAM even higher, DRAM would not be affected by the operations of the SoC.

A defense not requiring physical changes or sensors would be to prevent deactivation of the self-refresh of DRAM. For example, the control of self-refresh can be disabled by forbidding access rights to the DRAM refresh related register, even to users with root privileges. However, this may prevent useful features such as DRAM PUFs that rely on disabling the refresh. Further, DRAM refresh may be disabled in power saving modes, so such modes would have to be deactivated as well.

7. Conclusion

In this work, we presented new covert channels based on the bit flips of PoP DRAM and fingerprints of PoP DRAM PUF that can be implemented in commodity devices, without any hardware changes. We evaluated possibilities for data exfiltration based on how programs running on the cores in SoC can heat up DRAM. The evaluation of the PoP DRAM and covert channels showed their feasibility and high reliability. Consequently, this work showed that while PoP configuration saves space and makes designs more compact, it introduces new security risks. The code used in this work will be published open-source at http://caslab.csl.yale.edu/code/popchannels/ and can be used with any Raspberry Pi B+ for testing the PoP channels.

Acknowledgments

We would like to thank Alyssa Rosenzweig for help with the firmware code. This work was supported by the Shenzhen Science, Technology and Innovation Commission grant 20170117 and National Natural Science Foundation of China grant 61571116. This work was also supported by United States' National Science Foundation grant 1651945.

References

[1] B. W. Lampson, "A Note on the Confinement Problem," *Communications of the ACM*, vol. 16, no. 10, pp. 613–615, 1973.

[2] C. Jie and G. Venkataramani, "CC-Hunter: Uncovering Covert Timing Channels on Shared Processor Hardware," in *International Symposium on Microarchitecture*, ser. MICRO. IEEE/ACM, 2014, pp. 216–228.

[3] Z. Wang and R. B. Lee, "Covert and Side Channels Due to Processor Architecture," in *Annual Computer Security Applications Conference*, ser. ACSAC, 2006, pp. 473–482.

[4] S. Tian and J. Szefer, "Temporal Thermal Covert Channels in Cloud FPGAs," in *Proceedings of the International Symposium on Field-Programmable Gate Arrays*, ser. FPGA, February 2019.

[5] G. O. Dyrkolbotn and E. Snekkenes, "A Wireless Covert Channel on Smart Cards (Short Paper)," in *Proceedings of the International Conference on Information and Communications Security*, ser. ICICS. Springer-Verlag, December 2006, pp. 249–259.

[6] H. Lee, Y. Im, and Y. Shin, "Comparative Study Of 3D Package Configurations In Power Delivery And Thermal Perspective," in *International Wafer Level Packaging Conference*, ser. IWLPC. IEEE, 2018, pp. 1–7.

[7] M. A. Islam, S. Ren, and A. Wierman, "Exploiting a thermal side channel for power attacks in multi-tenant data centers," in *Conference on Computer and Communications Security*, ser. CCS. ACM, 2017, pp. 1079–1094.

[8] R. J. Masti, D. Rai, A. Ranganathan, C. Müller, L. Thiele, and S. Capkun, "Thermal Covert Channels on Multi-core Platforms," in *USENIX Security Symposium*, ser. SEC. USENIX Association, 2015, pp. 865–880.

[9] T. Iakymchuk, M. Nikodem, and K. Kepa, "Temperature-based covert channel in FPGA systems," in *International Workshop on Reconfigurable Communication-centric Systems-on-Chip*, ser. ReCoSoC. IEEE, 2011, pp. 1–7.

[10] W. Xiong, A. Schaller, N. A. Anagnostopoulos, M. U. Saleem, S. Gabmeyer, S. Katzenbeisser, and J. Szefer, "Run-time Accessible DRAM PUFs in Commodity Devices," in *Proceedings of the Conference on Cryptographic Hardware and Embedded Systems*, ser. CHES, August 2016.

[11] A. Schaller, W. Xiong, N. A. Anagnostopoulos, M. U. Saleem, S. Gabmeyer, B. Skoric, S. Katzenbeisser, and J. Szefer, "Decay-Based DRAM PUFs in Commodity Devices," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 3, pp. 1–14, 2019.

[12] W. Xiong, N. A. Anagnostopoulos, A. Schaller, S. Katzenbeisser, and J. Szefer, "Spying on Temperature using DRAM," in *Proceedings of the Design, Automation, and Test in Europe*, ser. DATE, March 2019.

[13] S. Rosenblatt, S. Chellappa, A. Cestero, N. Robson, T. Kirihata, and S. S. Iyer, "A Self-Authenticating Chip Architecture Using an Intrinsic Fingerprint of Embedded DRAM," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 11, pp. 2934–2943, 2013.

[14] F. Tehranipoor, N. Karimian, W. Yan, and J. A. Chandy, "DRAM-Based Intrinsic Physically Unclonable Functions for System-Level Security and Authentication," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 3, pp. 1085–1097, 2017.

[15] MathWorks, "Envelope," https://www.mathworks.com/help/signal/ref/envelope.html.

[16] M. Hiller, D. Merli, F. Stumpf, and G. Sigl, "Complementary IBS: Application specific error correction for PUFs," in *International Symposium on Hardware-Oriented Security and Trust*, ser. HOST. IEEE, 2012, pp. 1–6.

[17] J. Delvaux, D. Gu, D. Schellekens, and I. Verbauwhede, "Helper Data Algorithms for PUF-Based Key Generation: Overview and Analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 6, pp. 889–902, 2015.

[18] B. Alex, R. Alyssa, and B. Kristina, "rpi-open-firmware," https://github.com/christinaa/rpi-open-firmware.

[19] A. Agne, H. Hangmann, M. Happe, M. Platzner, and C. Plessl, "Seven recipes for setting your FPGA on fire A cookbook on heat generators," *Microprocessors Microsystems*, vol. 38, no. 8, pp. 911–919, 2014.