Spying on Temperature using DRAM

Wenjie Xiong[*], Nikolaos Athanasios Anagnostopoulos[†], André Schaller[†], Stefan Katzenbeisser[†], and Jakub Szefer[*]
[*]Yale University, New Haven, CT, USA [†]Technische Universität Darmstadt, Darmstadt, Germany
wenjie.xiong@yale.edu, {anagnostopoulos, schaller, katzenbeisser}@seceng.informatik.tu-darmstadt.de, jakub.szefer@yale.edu

***Abstract*—Today's ubiquitous IoT devices make spying on, and collecting data from, unsuspecting users possible. This paper shows a new attack where DRAM modules, widely used in IoT devices, can be abused to measure the temperature in the vicinity of the device in order to spy on a user's behavior. Specifically, the temperature dependency of the DRAM decay is used as a proxy for user's behavior in the vicinity of the device. The attack can be performed remotely by only changing the software of an IoT device, without requiring hardware changes, and with a resolution reaching $0.5°$C. Potential defenses to the temperature spying attack are presented in this paper as well.**

***Index Terms*—Security, IoT, DRAM, Temperature**

I. Introduction

Today, deployment of loT devices has become popular due to their low cost and ability to perform many useful functions. These devices are deployed in a variety of settings, such as homes and factories. However, this trend also gives malicious attackers a chance to gather information remotely, once they are able to compromise an IoT device, as most IoT devices are connected to the Internet. To defend against potential threats that arise due to IoT devices, there are increased efforts to ensure that the software of the IoT devices will handle information generated from the sensors properly, e.g., [1]–[3]. For example, to protect acoustic signals which may contain sensitive information, the software is prevented from leaking information collected from microphones [4].

However, even if the information from traditional sensors is handled properly, it has been shown that data can also be collected from other components or sensors of the devices, which were not considered before. For example, a gyroscope can be abused to measure acoustic signals [4]. Previously, the gyroscope measurements were considered to be only related to motion, meanwhile, researchers found a way to abuse them to gather sound recordings and to recognize speech. Also, the power usage of a mobile phone can be used to track location [5], while the power usage information was considered harmless before. As the ability for each component to collect information is not understood well, these attacks are not protected, putting users' privacy at risk.

This paper reports a new attack that leverages DRAM modules in IoT devices to spy on users. In particular, DRAM modules can be abused to monitor the ambient temperature around the device. Many IoT devices such as Raspberry Pi [6], Samsung ARTIK [7], or Intel Galileo [8], come with DRAM modules. Meanwhile, ambient temperature contains much critical information, e.g., if measured at the victim's home, it can reveal the victim's daily routine; if measured at a production line, it can reveal the temperature of the manufacturing process of a product; if measured at a data center, it can reveal the activity of the tenants [9].

This paper shows that it is practical to measure the ambient temperature with DRAM modules, by leveraging the DRAM decay behavior when the DRAM refresh is disabled. It has been shown that the DRAM decay depends on the ambient temperature [10]–[12]. This paper demonstrates that the attacker can thus map the DRAM decay results to temperature changes without physical access or having to measure the device at different known temperatures in advance. An attacker who is able to compromise an IoT device remotely, can launch the attack by using the DRAM decay behavior to spy on users, with a temperature resolution as good as $0.5°$C. The goal of this paper is to understand the potential threats that these DRAM modules introduce, and the countermeasures that should be taken by the manufacturers and users of IoT devices.

A. Paper Contributions

The contributions of the paper are as follows:

- This paper demonstrates that DRAM decay can be used to measure the ambient temperature via only software changes in IoT devices. The attacker can practically conduct the DRAM decay enrollments at a constant ambient temperature to later map the DRAM decay measurement results to different temperatures and guess user's behavior or environmental changes.
- The temperature resolution is shown to be as good as $0.5°$C in commodity, off-the-shelf IoT devices, enabling attackers to measure fine-grained temperature changes around the IoT devices.
- A set of defenses are discussed, such as limiting the software's ability to control the DRAM refresh.

II. DRAM Decay Characteristics

DRAM is one of the most widely used memories in computer devices. In DRAM, each bit of data is stored in a DRAM cell that consists of an access transistor and a capacitor, as shown in Fig. 1 (a). Each capacitor of the DRAM cell has two states, charged and dis-charged, which are used to store one bit of data.

DRAM is a volatile memory, and the stored data will be lost if the refresh or power are turned off. This is because capacitors lose charge over time. Fig. 1 (a) shows possible paths through which the charge on each capacitor can leak. To

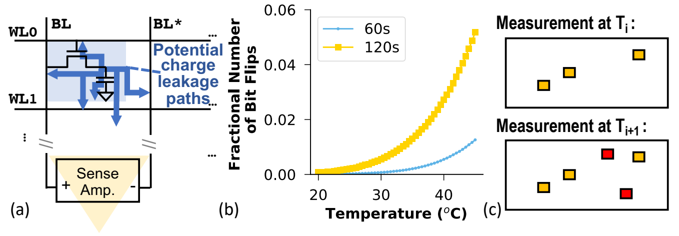

Fig. 1. (a) DRAM cell schematic. (b) Temperature dependency of fractional bit flips for DRAM modules from a tested Intel Galileo board. (c) Illustration of a sample DRAM array decay measurement at temperatures T_i and T_{i+1}, both with same decay time t; highlighted cells are the cells where a bit flip occurs – for the same decay time, more cells flip at the higher temperature (illustrated by red cells).

prevent data loss, each DRAM cell requires a periodic refresh, usually every 32ms.

When DRAM refresh is disabled, cells in charged state will steadily lose charge. If a charged cell loses enough charge, it becomes dis-charged and the stored data bit flips. The loss of charge over time is referred to as *DRAM decay*. The time a cell can keep the bit value without refresh is called the *retention time*. Different DRAM cells have different retention times. Thus, if the refresh is disabled for a longer time period, more cells' retention times are exceeded and more bit flips appear. Meanwhile, if a cell is initialized to a dis-charged state, its value will never flip as there is no charge to leak.

A DRAM decay measurement is a measurement showing which DRAM cells have flipped in a DRAM region after a given decay time t has elapsed. To perform a DRAM decay measurement, first, a DRAM region is selected and the cells in this region are initialized to a known value. For example, this work uses logical 0 as the initial value of all the cells[1]. The DRAM region is then allowed to decay, i.e., the refresh operation is disabled, for time t. After time t elapsed, the DRAM region is read, to observe which cells have flipped their value from initial logical 0 to logical 1 – these cells are the ones which have decayed during time t. Moreover, in an independent field of study, DRAM decay is leveraged for creating a PUF (Physical Unclonable Function), which can be used as a security primitive for authentication and key storage, e.g., [11]–[17].

The DRAM decay in particular depends on the temperature [10]–[12], and a higher temperature accelerates the charge leakage. The fractional number of bit flips (i.e., the number of bit flips in the DRAM region divided by the size of this region) for one Intel Galileo board [8] is shown in Fig. 1 (b). The number of bit flips increases as the temperature increases. Also, using a decay time of $t = 120$s will result in more bit flips than using a shorter decay time of $t = 60$s, for example.

[1]Note that some DRAM cells map logical 0 to the charged state, while others map logical 1 to the charged state. The exact mapping is not published by DRAM manufacturers, but we have empirically derived that about half the cells in the tested DRAM modules map logical 0 to the charged state. The cells that map logical 0 to dis-charged state simply do not contribute any value to the measurement, but also do not interfere with it.

Fig. 1 (c) illustrates example DRAM decay results for same decay time t, but at temperatures T_i and T_{i+1} ($T_{i+1} > T_i$). More bit flips appear at the higher temperature T_{i+1}, compared to the lower temperature T_i. Also, bit flips that occur at a lower temperature are a subset of bit flips that occur at a higher temperature.

III. Temperature Spy Attack

Many IoT devices may not have a dedicated temperature sensor – this work shows that even in absence of a temperature sensor, attackers can still leverage DRAM cells in the IoT devices to learn the ambient temperature.

The attacker first needs to compromise the remote IoT device to be able to control the DRAM refresh. Usually, he or she needs to compromise the kernel to measure the DRAM decay. Many IoT devices are vulnerable to exploits that can give kernel privileges.

Once the device is compromised, the attacker needs to take m enrollments. We show that the enrollments can be taken using different decay times at a constant, but possibly unknown, temperature. Based on the enrollments, the attacker can map the DRAM decay results to temperature. Then the attacker simply performs one DRAM decay measurement and learns the temperature.

A. Enrolling DRAM Decay at a Constant Temperature

To map a DRAM decay measurement of a DRAM region on a particular DRAM module to an ambient temperature, a set of measurements is needed, which are called the *enrollments*. Normally, this set of enrollments should be taken at a fixed decay time t and at m different temperatures $\{T_0, T_1, T_2, ..., T_m\}$ covering the temperature range of interest. However, it is not practical for an attacker to always control the ambient temperature for enrollment.

To overcome the challenge, this work shows that the attacker can enroll the DRAM at one constant temperature instead of m different temperatures. This is possible because the DRAM decay at different temperatures can be simulated by measurements with different decay times at a constant temperature. For example, the decay result of decay time $2t$ at enrollment temperature T is similar to the decay result of decay time t at temperature $T{+}10$°C. The relationship between the temperature and time is further derived and evaluated in Section IV-B. In this way, the attacker takes enrollments at a constant temperature T_0 for decay times $\{t_0, t_1, t_2, ..., t_m\}$ to simulate decay results at $\{T_0, T_1, T_2, ..., T_m\}$ for decay time t_0 at each of these temperatures. In Section IV-B, we experimentally validate the approach.

Due to the manufacturing variations of each DRAM module, the enrollment should be taken on the same device as the one used later to spy on the users. To do this, the attacker can conduct enrollment when he or she assumes that the ambient temperature is constant, e.g., it is reasonable to assume the temperature will not change within the enrollment time in a smart home at night.

B. Mapping DRAM Decay to Temperature Changes

Given the m enrollments simulating different temperatures, a mapping between the DRAM decay results and the ambient temperatures can be generated by counting the number of bit flips in the i-th enrollment and mapping that number of bit flips to temperature T_i. Later, given a DRAM decay measurement, the number of bit flips can be counted and compared with the enrollment measurements. The temperature of the measurement is seen to be the same as the temperature of the enrollment measurement with the most similar number of bit flips. However, when counting the bit flips, the whole DRAM region measured needs to be processed. This introduces memory bandwidth and computational overhead at measurement time.

To overcome this challenge, we propose to only use *indicator cells*, which is a subset of cells in the DRAM region, to reduce the overhead. With enrollments simulating temperature T_i and T_{i+1} ($T_{i+1} > T_i$), to choose the indicator cells for T_i, the two enrollments are compared, and cells which flip at T_{i+1} but not T_i are the *candidate indicator cells*. For example, candidate indicator cells are highlighted in red in Fig. 1 (c). Among the candidate cells, l cells are selected as *indicator cells*. Depending on the expected noise level (see Section IV) the number l can be increased. Typically, an odd number of cells is needed to allow majority voting. If l candidate indicator cells cannot be identified, a larger DRAM region or a longer decay time t should be used. The locations of all the indicator cells for all temperatures need to be saved. Later, during a temperature spy attack, only the $l \times (m-1)$ indicator cells (indicator cells for all enrolled temperatures) need to be read. For each potential temperature, the majority vote of l indicator cells is used to decide whether the current temperature is above T_i. Then after $(m-1)$ majority votes, the current temperature $T_{current}$ is known. This can save memory bandwidth by orders of magnitude because only dozens of indicator cells instead of all the KiBs or MiBs of cells in the DRAM region are measured at measurement time.

C. DRAM Decay Measurement at System Runtime

It is not trivial to make the DRAM decay measurement at system runtime without hardware changes, because it is not possible to disable the DRAM refresh for arbitrary DRAM regions. If the whole DRAM module's refresh is disabled, all content of the memory will eventually decay and errors in the memory contents will cause the system to crash. As a solution, similar to the approach of [11], this work uses a kernel module to disable the refresh of the whole DRAM module while issuing extra memory accesses to the memory regions holding the critical system data. Each DRAM access also behaves as a refresh, so the system data that is explicitly accessed will not decay. At the same time, the other cells in DRAM which are not accessed will decay.

IV. Evaluation

The evaluation is conducted on Intel Galileo Gen 2 [8] IoT development boards, which have an Intel Quark X1000 SoC, with two 128MiB DDR3 from Micron. A kernel module is loaded to measure the DRAM decay in the chosen DRAM region. In total, four Intel Galileo boards are measured.

First, Section IV-A evaluates the resolution of DRAM as a temperature sensor. Then, Section IV-B shows that it is possible to take enrollments at a constant temperature. In Sections IV-C and IV-D, attack examples are presented and the complexity of the attack is discussed.

A. DRAM as a Temperature Sensor

To show that DRAM decay can be used to observe the device's ambient temperature, this section answers the following questions: (1) What level of error rate in the measurements can be corrected by the majority voting of l indicator cells? (2) How many candidate indicator cells are available for a given DRAM region size in the tested devices? (3) Are the chosen indicator cells reliable? (4) How sensitive is DRAM decay to temperature changes?

To allow for precise evaluation of the temperature-dependent characteristics of DRAM modules, a TestEquity 1007C thermal chamber [18] is used to control the ambient temperature.

Supported Measurement Error Rates. When the attacker attempts to derive the temperature from the DRAM decay measurements, he or she will use l indicator cells and perform majority votes. The minimum value of l is 3. With $l = 3$, an error rate of up to 33% can be corrected by the majority vote. With $l = 5$, the majority vote can correct error rate of 40%, and so forth. In practice, the attacker can set l based on the noise and the number of available candidate indicator cells in the DRAM region.

Number of Available Candidate Indicator Cells. The DRAM decay measurements were performed at $\mathcal{T} = \{20, 21, ..., 45\}$°C, where $dT = T_{i+1} - T_i$ denotes the step between temperature points in $\mathcal{T}$, thus here $dT = 1$°C. Fig. 2 shows the number of bit flips at temperature T_{i+1} but not T_i, i.e., candidate indicator cells. The results are the average of DRAM regions on four Galileo boards. Two decay times of $t = 60$s and $t = 120$s were tested, with DRAM region sizes of 512KiB, 1MiB, and 2MiB. The number of candidate indicator cells depends on the DRAM region size and the decay time t. Within the range, the smallest number of candidate indicator cells occurs when the attacker measures a 512KiB DRAM region at 20°C – one of the tested boards gives only 2 indicator cells. However, 1MiB DRAM is sufficient to support $l = 3$ or $l = 5$, and thus support error rates of up to 40%.

Reliability of Indicator Cells. An ideal indicator cell for T_i should never flip at T_i and always flip at T_{i+1}. To evaluate the reliability, at each temperature five measurements are taken for each Galileo board. The first measurement is used as an enrollment, and the other four spy measurements are used for testing the reliability. Fig. 3 and Fig. 4 show the average and maximum Bit Error Rate (BER) of the spy measurements. The BER for each temperature T_i is calculated as follows: (i) in the enrollment measurement, the number of candidate

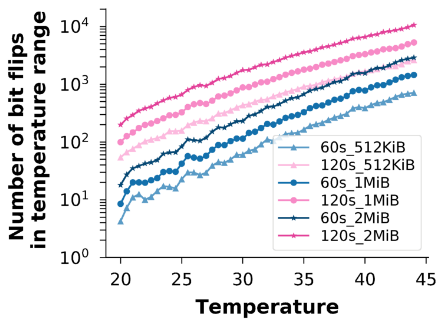

Fig. 2. Number of bit flips in temperature range $[T, T{+}1^\circ C]$ versus temperature T for different DRAM region sizes.

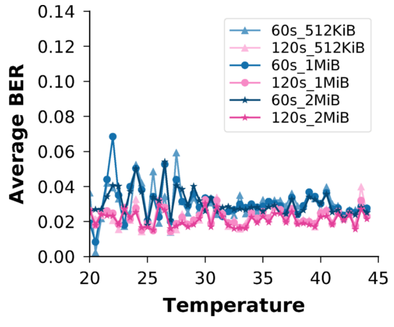

Fig. 3. Average BER in temperature range $[T, T{+}1^\circ C]$ versus temperature T for different DRAM region sizes.

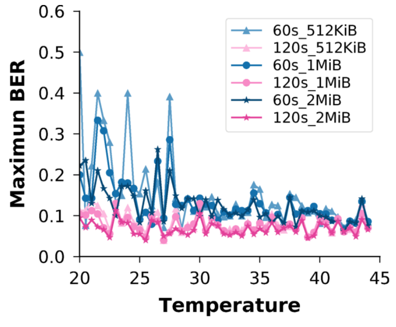

Fig. 4. Maximum BER in temperature range $[T, T{+}1^\circ C]$ versus temperature T for different DRAM region sizes.

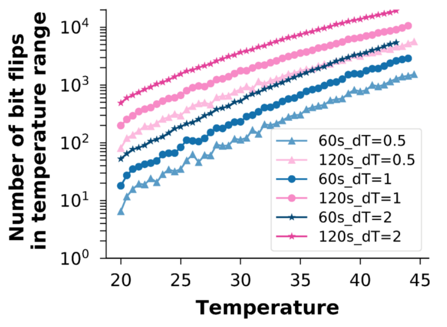

Fig. 5. Number of bit flips in temperature range $[T, T{+}dT]$ versus temperature T for different dT values in 2MiB DRAM region.

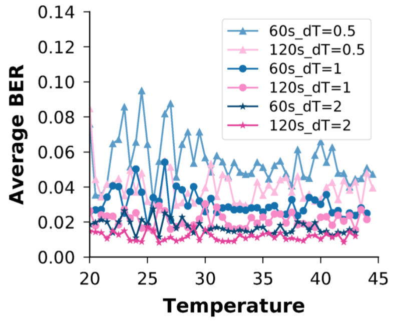

Fig. 6. Average BER in temperature range $[T, T{+}dT]$ versus temperature T for different dT values in 2MiB DRAM region.

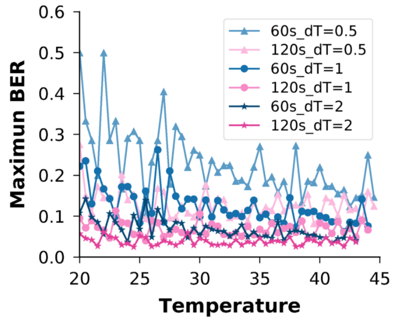

Fig. 7. Maximum BER in temperature range $[T, T{+}dT]$ versus temperature T for different dT values in 2MiB DRAM region.

indicator cells are counted; (ii) in the spy measurement, if an indicator cell flips at T_i or an indicator cell does not flip at T_{i+1}, this cell is seen as an error; (iii) the number of errors is counted and divided by the result from step (i) to compute the BER. Fig. 3 and Fig. 4 show the average and maximum BER across the four boards and four spy measurements. As shown in Fig. 3 and Fig. 4, a longer decay time, a larger DRAM region size, or a higher temperature will result in a smaller BER. This is because a longer decay time, a larger DRAM region, and a higher temperature yield more indicator cells (and more reliable indicator cells). The average BER is much smaller than the maximum case, meaning that there are only a few cases where the BER is high. As shown in Fig. 4, to obtain reliable results, at least 1 MiB DRAM region size is needed to achieve a BER of less than 33% for $l = 3$.

Temperature Sensitivity. We further explored different temperature resolutions, where $dT = 0.5°\text{C}, 1°\text{C}, 2°\text{C}$ separately. The results in Fig. 5, Fig. 6 and Fig. 7 are retrieved from all four Galileo boards with a DRAM region of 2MiB and a decay time of either 60s or 120s. The data are processed in the same way as in Fig. 2 – 4. As shown in Fig. 5, at least 5 indicator cells can be found in 2 MiB regions in the temperature range $[T_i, T_i + dT]$ for all T_i and dT considered. Fig. 7 shows that when decay time t = 120s and dT = 0.5°C, the maximum BER can still be corrected by the majority vote of $l = 3$ cells. However, with decay time t = 60s and dT = 0.5°C, the maximum BER is too large to be corrected by the majority vote of $l = 3$ cells, a larger DRAM region or longer decay time should be used.

B. Enrollments at a Constant Temperature

Here, we show that the enrollments can be taken at a fixed temperature with different decay times, and the attacker can, from these enrollments, derive the expected DRAM decay at other temperatures, even if he or she never enrolled the device at these temperatures. In particular, we show how the DRAM decay with decay time t_{real} and temperature T_{real} can be simulated by a measurement with decay time t_{sim} and temperature T_{sim}.

We denote $\Delta T_{rs} = T_{real} - T_{sim}$, which is the temperature difference between the real temperature (T_{real}), and the temperature that the attacker wants to enroll to simulate the real temperature (T_{sim}). As indicated in [11], [13], the DRAM decay time t_{sim} and t_{real}, and temperature ΔT_{rs} have the following relationship:

$$t_{sim} = t_{real} \times e^{k \Delta T_{rs}}. \quad (1)$$

Same models of DRAM chips have the same temperature index k, so the attacker can compute k using his or her own device (where he or she can control the temperature), then use that k for the attack on a remote device.

To validate Equation (1), the simulation measurements are taken at T_{sim} = 25°C and T_{sim} = 30°C, to simulate the

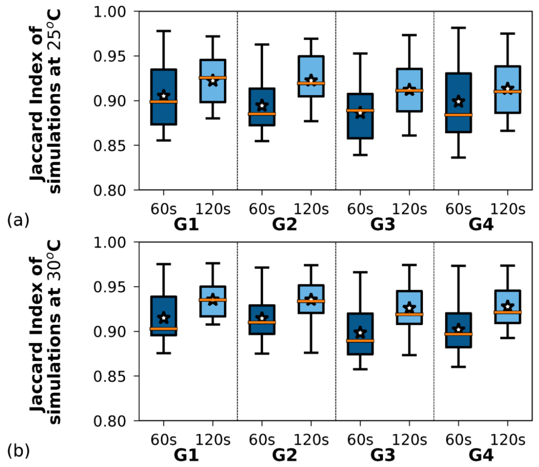

Fig. 8. Jaccard Index between simulation measurements at (a) 25°C and (b) 30°C and real measurements with decay time of 60s or 120s.

DRAM decay at temperature range $T_{real} = 20$ to 40°C and $T_{real} = 25$ to 45°C, respectively. The measurements are designed to simulate the decay time of both 60s and 120s. To simulate decay time $t_{real} = 60$s (120s), ten different decay times in the range of $t_{sim} = 45$s to 160s (90s to 320s) are measured.

To estimate the temperature index k, the simulation measurements and the real measurements are compared. For each simulation measurement, we find the real measurement that has the most similar number of bit flips, and record the pair t_{sim} and $\Delta T'_{rs}$. With the pairs of t_{sim} and $\Delta T'_{rs}$ across $t_{real} = 60$s or 120s, $T_{sim} = 25$°C or 30°C, and four Intel Galileo boards, the best fit temperature index k can be computed. The resulting k is 0.07.

To show that the simulation measurements are similar to the real measurements, using k, we compute the ΔT_{rs} for each t_{sim}. We then compare each pair of real measurement (t_{real}, T_{real}) and simulation measurements (t_{sim}, T_{sim}). To compare the two measurements, we use the *Jaccard Index* [19]. Let $\mathcal{R}$ and $\mathcal{S}$ denote the set of bit flips in the real measurement and simulation measurement respectively. Jaccard Index is calculated by $J = \frac{|\mathcal{S} \cap \mathcal{R}|}{|\mathcal{S} \cup \mathcal{R}|}$. If the resulting J is close to 1, it means the two measurements are very similar.

Fig. 8 shows the distribution of Jaccard Index for $t_{real} =$ 60s or 120s for all four Intel Galileos. Each box contains ten different simulation decay times and the corresponding real temperatures. The stars indicate the average of the data set, the orange bars indicate the median. The Jaccard Index is larger than 0.85, indicating that the simulation measurements and the real measurements are very similar. Thus, the enrollments can be taken at a fixed temperature to cover a range of different temperatures.

C. Attacks in Practice

To show the attack is practical, we deployed several Intel Galileo boards in the open air in two rooms and in a server rack. A bare Yocto Linux kernel is running during the test. We measured the DRAM decay of 60s with 2MiB DRAM region every 5 minutes to infer the ambient temperature. For each of the boards, enrollments with decay times ranging from 50s to 75s (in step of 1s) are taken. According to Equation (1) and $k = 0.07$, the enrollments can simulate the ambient temperature change of $[-3^o\text{C}, +3^o\text{C}]$. Note that in total 26 enrollments are taken for the temperature range, so the actual temperature resolution is better than 0.5^oC. A thermocouple is used to get the actual temperature during the enrollment for reference. Indicator cells are generated based on the enrollment, and later used to map the decay results to temperatures. More than twenty candidate indicator cells are found in 2MiB for each temperature, so $l = 21$ is used.

Fig. 9 (a) and (b) show the temperature in two different rooms measured by the DRAM and a thermocouple for 24 hours. The temperatures measured by the DRAM match the results of the thermocouple. As shown in the figure, during the night, the temperature is stable; while during the day, due to human activities, the temperature fluctuates in both rooms. Thus, if the attacker can monitor the temperature, this puts the victim's privacy at risk.

In the second experiment, we deployed two Galileo boards in a server rack. Fig. 9 (c) shows the temperature measured by two DRAM modules, DRAM1 and DRAM2. DRAM1 is located closer to fans. The thermocouple is placed near DRAM1. The arrows show when the server starts to run a job for 25 minutes. When the server runs, it will gradually heat DRAM2. Subsequently, the fan starts working, and ambient temperature drops (especially for DRAM1 and the thermocouple). Consequently, using only the IoT device's DRAM,

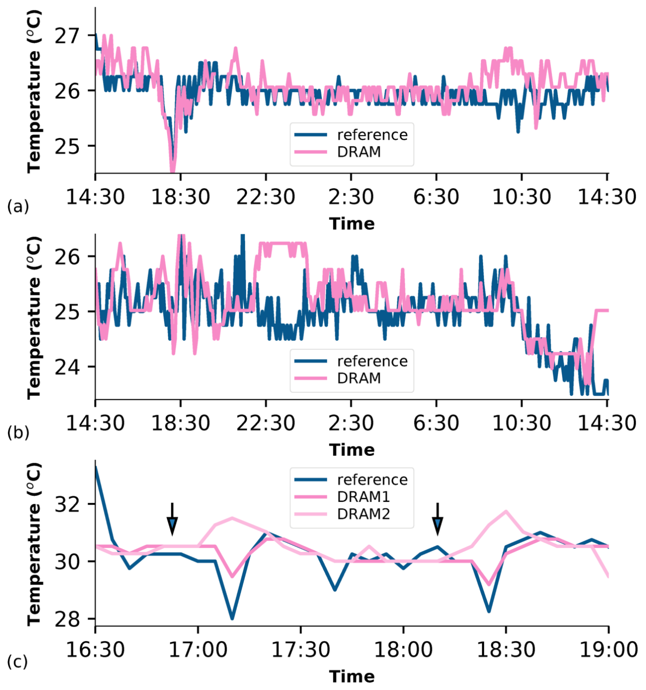

Fig. 9. (a), (b) Results of measuring the temperature every 5 minutes with DRAM module in two different rooms for 24 hours. (c) Results of measuring the temperature every 5 minutes with DRAM modules in a server rack. The arrows show when the server starts to run a job.

the attacker can monitor the temperature change of the server, which could create a side-channel to reveal the activity of the tenants [9].

D. Attack Complexity

The attacker's efforts consist of taking the enrollments and conducting each temperature readout. For both, the attacker needs to be able to run malicious kernel code on the victim platform to control the DRAM refresh.

The enrollment time consists of measurement time, data transfer time and time to identify indicator cells. The measurement time is the decay time plus the time to initialize (write) and to read from the DRAM region. For a 2MiB memory region, on Intel Galileo, it takes about half a minute to read or write the region. So one enrollment takes about two minutes considering a decay time of $t = 60$s: half a minute to initialize, one minute to allow decay to happen and half a minute to read the DRAM region to locate the decayed bits. The total number of measurements depends on the temperature range and temperature resolution required. To take ten enrollments, assuming an average enrollment decay time of $t = 60$s, it takes less than half an hour. The data transfer time depends on the size of data to transfer and the network speeds. The time to compare the enrollment measurement and identify the indicator cells are negligible.

The temperature readout time consists of a single measurement. Furthermore, because only the indicator cells need to be measured, the time to initialize and read the result is negligible compared to the decay time.

V. Countermeasures

One simple way to mitigate the temperature spy attack is to prevent disabling of the DRAM refresh, as the attacker needs to disable the DRAM refresh to measure the DRAM decay. Since disabling the DRAM refresh can only be achieved in the kernel space in almost all platforms, the attacker has to inject untrusted code into the kernel or firmware. One countermeasure is to protect the kernel and firmware code.

However, forcing the DRAM refresh to be always on is not desired from an energy saving perspective. Because a memory deep sleep mode usually exists, where the DRAM refresh is off, an attacker can write initial values into the DRAM region and force DRAM into sleep mode, so that DRAM decays. To prevent this attack, the system needs to always zero out the whole memory when DRAM wakes up.

VI. Conclusion

In this paper, we demonstrated how the widely used DRAM modules can be abused to act as a temperature spy in IoT devices. We showed how the attacker only needs to modify the software of a device, take enrollment measurements at a constant temperature, and then, can monitor the ambient temperature by measuring the DRAM decay. Moreover, the attack has a high temperature resolution. This attack warns us that DRAM components in IoT devices pose potential threats and countermeasures should be taken. The attack and the analysis code is available under open-source license at http://caslab.csl.yale.edu/code/tempspy/. Future work should examine whether this could lead to a thermal side channel attack to extract cryptographic keys.

Acknowledgment

This work has been supported by NSF through grant #1651945 and by DFG as part of the project P3 within the CRC 1119 CROSSING.

References

[1] P. Koeberl, S. Schulz, A.-R. Sadeghi, and V. Varadharajan, "TrustLite: A security architecture for tiny embedded devices," in *Proceedings of the Ninth European Conference on Computer Systems*. ACM, 2014.

[2] F. Kohnhäuser, A. Schaller, and S. Katzenbeisser, "PUF-based software protection for low-end embedded devices," in *International Conference on Trust and Trustworthy Computing*. Springer, 2015, pp. 3–21.

[3] M. Potkonjak, S. Meguerdichian, and J. L. Wong, "Trusted sensors and remote sensing," in *Sensors, 2010 IEEE*. IEEE, 2010, pp. 1104–1107.

[4] Y. Michalevsky, D. Boneh, and G. Nakibly, "Gyrophone: Recognizing speech from gyroscope signals." in *USENIX Security Symposium*, 2014, pp. 1053–1067.

[5] Y. Michalevsky, A. Schulman, G. A. Veerapandian, D. Boneh, and G. Nakibly, "Powerspy: Location tracking using mobile device power analysis." in *USENIX Security Symposium*, 2015, pp. 785–800.

[6] "Raspberry Pi," https://www.raspberrypi.org/, accessed July 2018.

[7] "Samsung ARTIK," https://www.artik.io/, accessed July 2018.

[8] "Intel® Galileo Gen 2," https://ark.intel.com/products/83137/Intel-Galileo-Gen-2-Board, accessed July 2018.

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

[10] J. Liu, B. Jaiyen, Y. Kim, C. Wilkerson, and O. Mutlu, "An experimental study of data retention behavior in modern DRAM devices: Implications for retention time profiling mechanisms," in *ACM SIGARCH Computer Architecture News*, vol. 41, no. 3. ACM, 2013, pp. 60–71.

[11] 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*, August 2016, pp. 432–453.

[12] A. Schaller, W. Xiong, N. A. Anagnostopoulos, M. U. Saleem, S. Gabmeyer, S. Katzenbeisser, and J. Szefer, "Intrinsic rowhammer PUFs: Leveraging the rowhammer effect for improved security," in *International Symposium on Hardware Oriented Security and Trust*. IEEE, 2017, pp. 1–7.

[13] 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*, 2018.

[14] S. Rosenblatt, D. Fainstein, A. Cestero, J. Safran, N. Robson, T. Kirihata, and S. S. Iyer, "Field tolerant dynamic intrinsic chip ID using 32 nm high-k/metal gate SOI embedded DRAM," *IEEE Journal of Solid-State Circuits*, pp. 940–947, 2013.

[15] 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*, pp. 2934–2943, 2013.

[16] A. Rahmati, M. Hicks, D. E. Holcomb, and K. Fu, "Probable cause: The deanonymizing effects of approximate DRAM," in *Proceedings of the International Symposium on Computer Architecture*. ACM / IEEE, 2015, pp. 604–615.

[17] S. Sutar, A. Raha, and V. Raghunathan, "D-PUF: An intrinsically reconfigurable DRAM PUF for device authentication in embedded systems," in *International Conference on Compilers, Architectures, and Synthesis of Embedded Systems*. IEEE, 2016, pp. 1–10.

[18] "Testequity 1007c temperature chamber," https://www.testequity.com/products/598/ accessed Jul. 2018.

[19] P. Jaccard, *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.