JULIET-PUF: Enhancing the Security of IoT-Based SRAM-PUFs Using the Remanence Decay Effect

Amit Kama*, Michael Amar*, Snir Gaaton, Kang Wang, Yifan Tu, and Yossi Oren, *Senior Member, IEEE*

***Abstract*—The cloud-based Internet of Things (IoT) enables the creation of innovative computer applications based on sensing, analyzing, and controlling the physical world. IoT deployments, however, are at a particular risk of counterfeiting, through which an adversary can corrupt the entire ecosystem. Therefore, entity authentication of edge devices is considered an essential part of the security of IoT systems. This research addresses the challenge of generating a unique ID in IoT devices. Unique IDs allow the IoT system maker to identify each edge device, and to ensure that only genuine devices can upload data to the cloud. Traditional ID mechanisms are not feasible in IoT, due to the edge device's constrained runtime environment, or the additional costs and the deployment difficulties that they introduce. In this work, we present JULIET-PUF, a novel PUF-based method for IoT identification, which relies on SRAM content retrieval after power glitches with time differences. Our scheme comes with no added hardware cost on the edge device. We evaluate JULIET-PUF using a dataset of 24 units of a popular commercial IoT device, and show that it is on average 95.58 times more secure than the common use of SRAM-PUF.**

***Index Terms*—Physical security, Entity authentication, Physically Unclonable Functions, SRAM-PUF.**

I. INTRODUCTION

INTERNET of Things systems are typically made up of a large set of low-cost devices, which are connected to a powerful cloud server. The added value of the system derives from a combination of the two elements: the edge devices provide sensing and actuation capabilities, and the cloud server makes intelligent decisions based on the aggregated data, and provides convenient access to the edge devices through a centralized platform. For example, a distributed network of security cameras can each monitor an individual resident's doorway, and the centralized cloud server can aggregate this data to derive higher-level insights about the conditions in the entire neighborhood.

When a consumer purchases an IoT edge device, the amount paid for the device covers not only the relatively low cost of manufacturing the edge device itself, but also the cost of developing and maintaining the expensive cloud server. This cost structure makes the IoT ecosystem particularly vulnerable to counterfeiting. For example, an unscrupulous vendor can reverse-engineer a competitor's edge device and create a low-cost clone which takes advantage of the competitor's cloud service; since this unscrupulous vendor spends nothing on the cloud service, he/she can offer a counterfeit device at a much lower cost than the original, while offering the same functionality. This risk is exacerbated by the reliance of many IoT manufacturers on original equipment manufacturers (OEMs), who create the edge devices as subcontractors. Since these OEMs already possess the hardware and software specifications of the edge device, they can simply produce more edge devices than that requested by the IoT manufacturers and sell the extra units at a discount on the counterfeit market. The relative low cost and simplicity of common edge devices also raises another risk. A vendor interested in causing harm to a competitor can create malicious edge devices that connect to the competitor's cloud server and feed it false data. This will cause the cloud server's algorithm to make incorrect decisions based on the corrupted data, and ultimately, it can also reduce consumer trust in the vendor's platform.

Because of these risks, it is important to properly authenticate edge devices before they are allowed to connect to the server. Essentially, the IoT system designer should maintain a list of edge devices authorized to interact with the server so that connection attempts of other devices will be rejected. This enables the vendor to ensure that only authorized devices benefit from the value provided by the powerful cloud server,

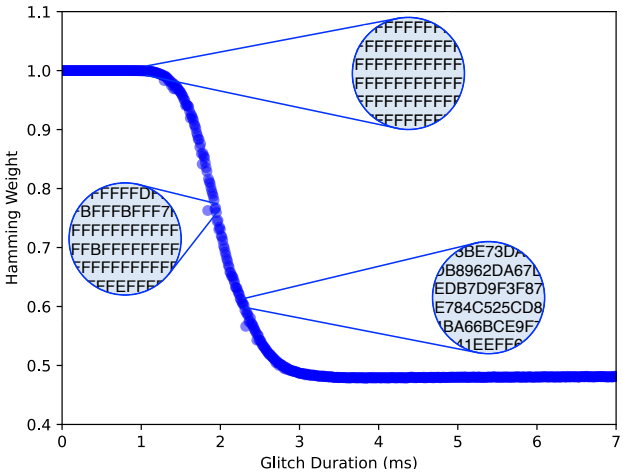

Fig. 1. The SRAM content of a device gradually changes over the various glitch durations, allowing multiple challenge-response pairs (CRPs) to be generated per device.

A. Kama, M. Amar, S. Gaaton and Y. Oren are with the Department of Software and Information Systems Engineering, Ben Gurion University of the Negev, Israel

K. Wang and Y. Tu are with Alibaba Group

* A. Kama and M. Amar contributed equally to this article.

Manuscript received TBD; revised TBD.

and allows the vendor to defend the cloud service from attacks such as the injection of malicious data.

A crucial requirement of any such authentication scheme is the ability to provide each device with a unique identifier. To prevent counterfeiting, this identifier should be difficult for counterfeiters to copy. Motivated by the cost sensitivity IoT edge devices, many studies have examined ways of providing this authentication ability without increasing the cost of the device. One promising method of assigning a unique identifier is to use intrinsic physically unclonable functions (PUFs). A PUF was defined by Barbareschi *et al.* in [1] as a function realized by means of a physical object, which relies on random variations introduced during manufacturing processes in order to be difficult to predict, yet easy to evaluate.

In general, to interact with a PUF, the authenticator typically issues the PUF with a *challenge*, by controlling some input of the PUF, such as the angle of illumination or the value of an input register. Then, the authenticator collects the *response* to this challenge, by measuring the effect of this challenge on the PUF's value. To use PUFs as part of an authentication scheme, the authenticator operates in two phases – an enrollment phase, and an authentication phase. In the *enrollment phase*, the authenticator issues a series of challenges to the PUF, and then records the responses of the PUF to each challenge and stores these challenge-response pairs (CRPs) in a central database. In the *authentication phase*, the authenticator issues a single challenge (from the set of stored CRPs) to the PUF, and the edge device extracts the PUF response itself and proves to the server that it knows the PUF response, potentially using a secure protocol. An example of such an authentication scheme can be seen in [2].

Since PUF responses are typically measurements of physical phenomena, they are affected by measurement artifacts, or noise. This makes them slightly different each time they are read out. To compensate for this, PUF response extraction usually involves an error correction step in which the raw response is processed using an algorithm such as Reed-Solomon [3] or LDPC [4], [5]. As discussed by Xiong *et al.* in [6], PUF responses should be as stable as possible when the same challenge is repeated (low intra-challenge distance), but they should also be significantly different than the responses provided for a different challenge (high inter-challenge distance).

One of the most commonly-used types of intrinsic PUFs is the SRAM-PUF. As defined by Cortez *et al.*, SRAM-PUFs take advantage of the way SRAM acts on startup. As each SRAM cell has its own bias toward a preferred startup value (one, zero, or random), a unique fingerprint can be created using multiple SRAM cells' startup values [7]. As noted by Mispan *et al.* [8], SRAM-PUFs are a natural match for IoT authentication. There are two main reasons for this good fit: virtually all microcontrollers make use of SRAM memory, and reading the response from the SRAM requires nothing more than reading from a large block of uninitialized memory. The main limitation of SRAM-PUFs is that they do not have an explicit challenge step. As a result, each device only has a single possible response – the content of the SRAM memory when the device is turned on. While over-the-air adversaries who monitor the communication between the device and the authenticator can be prevented from learning this secret response using methods such as zero-knowledge proofs [9], [10], IoT edge devices must be analyzed while taking into account their very low cost and high availability. It is relatively straightforward for an adversary to purchase an IoT device and read out the SRAM startup values, either by using a malicious software update or by directly interfacing with the SRAM in hardware. Once the adversary knows the startup value, it is trivial for him/her to create a cloned device that impersonates the IoT device by repeating the extracted SRAM-PUF response. Since there is no challenge, all the attacker needs to do to create such a counterfeit device is to store the cloned PUF response in a non-volatile memory. Thus, the cost of creating such a clone is minimal.

The objective of our work is to enhance the security of SRAM-PUFs in IoT deployments by introducing a challenge component, without adding any additional hardware to the authenticating circuit. The key point enabling this is the fact that different bits in SRAM have different remanence times, as originally observed by Holcomb [11]. An outcome of this observation, originally noted by Oren [12], is that if an SRAM device is powered off for a specified fixed brief amount of time (an act commonly referred to as *power glitching*), some of the bits in the SRAM will revert to their default, or PUF, states, while the rest will retain their original value, as illustrated in Figure 1. This observation serves as the basis of a challenge response scheme we call JULIET-PUF[1], a novel PUF-based unique ID generation method, where in a nutshell, the authenticator sets the bits of the SRAM to a fixed value, then powers off the device for a specified brief duration, and finally turns on the device and reads out the value of the SRAM, as illustrated in Figure 2. The glitch time is the implicit challenge: for each selected time, a different subset of the SRAM bits will revert to its PUF state, resulting in a different PUF response. To counterfeit such a device, an adversary would need to both record all possible responses of the device to different power glitch durations, and use a timekeeping mechanism that can measure the duration of the power glitch and map it to the correct stored response. These two additional requirements would significantly increase the cost of producing such a counterfeit, making it less cost-effective than purchasing the original device, as we show in Section IV-B. The JULIET-PUF scheme consists of two phases – the enrollment phase, which is the process of assembling various CRPs, and the authentication phase (also referred to as the authentication process) which works as follows. When entity authentication is required, the cloud server, as the authenticator, challenges the edge device and measures its response. As shown in Figure 2, the authentication process involves the following steps:

1) At the beginning of the authentication process, the edge device sends an authentication request to the cloud server, which includes its ID. The cloud server then randomly

[1]In Shakespeare's play Romeo and Juliet, the heroine Juliet drinks a sleeping potion in order to present a "borrow'd likeness of shrunk death" for a limited period of time.

chooses a challenge from the CRP list of that particular device.
2) The values of the SRAM cells of the edge device are set to one, and the device then shuts down for the duration of the power glitch. After being turned back on, the device sends its SRAM content as the response.
3) The cloud server receives the response and calculates the distance between the response and the response corresponding to the challenge, as specified in the CRP list. If the calculated distance does not exceed a certain threshold, the device is approved. Otherwise, it is rejected.

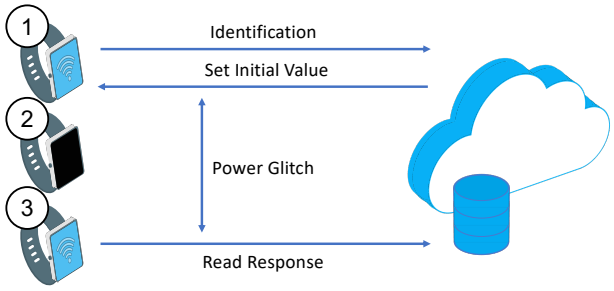

Fig. 2. JULIET-PUF authentication process.

As our results show, JULIET-PUF increases the challenge space of SRAM-PUFs by a factor of almost 100, without adding any extra hardware to the authenticating IoT device.

To summarize, the contributions of our paper are as follows:

- We introduce the JULIET-PUF scheme and its associated challenge-response protocol.
- We investigate the factors that affect the scheme's security and robustness, and show how it is influenced by the temperature, the accuracy of the timing source, and the amount of bits in the PUF response.
- We present a proof-of-concept demonstration of JULIET-PUF and report on its performance.
- We perform a security analysis and show that the proposed method improves the security level of SRAM-PUF schemes, especially in the attack model appropriate for IoT devices.

We believe that our proposed method can be used whenever SRAM-PUF schemes are used today, significantly increasing the security of SRAM-PUF IoT deployments without increasing their cost.

II. METHODS

The consistency of SRAM cell values after a power glitch is affected by three factors: the environmental temperature, the accuracy of the glitch period, and the intrinsic instability of the SRAM itself. We aim to develop a fingerprinting method with a reasonable authentication time and increased security, that takes these factors into account while meeting the cost constraints of the IoT environment.

We define *inter-distance* as the Hamming distance between multiple samples (PUF responses) of the same device, which are produced by power glitches with different durations, and *intra-distance* as the distance between samples of the same device, which are produced by power glitches with similar durations. To increase the security of our proposed method, we aim to significantly increase the number of SRAM-PUF CRPs. To do so, we want to make the intra-distance as low as possible, relative to the inter-distance.

In this section, we introduce the proposed methodology. A detailed explanation of the highly parallelized data collection facility we developed, which consists of 24 units of a popular commercial IoT device, is provided, as well as an explanation of the enrollment phase – the process of assembling CRPs, from trace collection through distance calculation, bit selection, challenge selection, and the final calculation of the representative PUF for each challenge. We then describe the authentication phase.

A. Development of an Innovative Data Collection Facility

In order to generate a large amount of CRPs and ensure that JULIET-PUF fulfills its objectives, a large quantity of high-quality data needs to be collected. To do so, we developed a highly parallelized data collection facility capable of profiling a large number of IoT devices simultaneously. The facility includes remote interfaces, such as power supply management, programming, and communication.

We chose to implement our proposed method on the nRF52832 SoC [13], a general-purpose multiprotocol SoC capable of handling demanding application and communication tasks quickly, which can be found in many IoT devices, including Apple AirTags, P8 smartphones and Casper glow lights. Our experimental environment consists of 24 units of an IoT device equipped with this SoC, specifically the NORDIC nRF52 development kit.

To ensure that the facility is suitable for implementing JULIET-PUF, we modified each of the 24 devices in such a way that allows us to perform power glitching on them, efficiently and inexpensively. We note that these adjustments do not affect the component cost of the devices.

We initially modified two potential types of devices, the NORDIC nRF52 DK and the Adafruit Bluefruit nRF52 Feather, in order to determine which type is more suitable for our experimental setup.

After examining the NORDIC device's power plan and filtering, we decided to cut the PCB track shorting a solder bridge to connect a transistor to the device's current measurement legs. By doing so, we were able to control the incoming voltage. We also removed some stabilizers, so they would not interfere with our measurement setup. The modified NORDIC nRF52 DK can be seen in Figure 3.

After examining the Adafruit Bluefruit nRF52 Feather device's power plan and filtering, which also contains the nRF52832 SoC, we concluded that the component that needs a modification is the linear voltage regulator, whose fifth leg is connected to a stabilizer, and through which the voltage continues to the chip. To control the power glitches, we lifted the regulator's fifth leg and connected a 2-pin header – one pin to the fifth leg and the other pin to the original connection

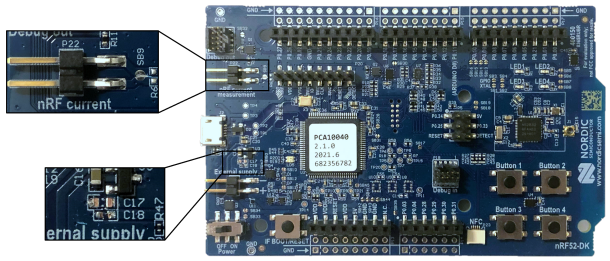

Fig. 3. Modified NORDIC nRF52 DK.

of the fifth leg to the track. By doing so, we were able to put a transistor on top of the header. We then removed some stabilizers, so they would not interfere. The modified Adafruit Bluefruit nRF52 Feather is presented in Figure 4.

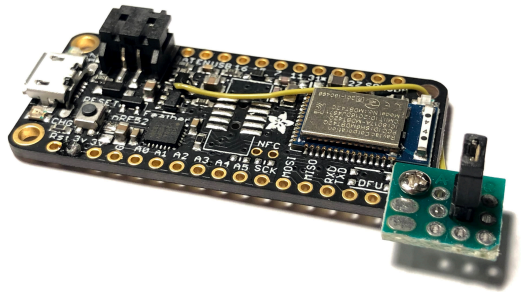

Fig. 4. Modified Adafruit Bluefruit nRF52 Feather.

After examining the modifications required for each of the devices, we decided to use the NORDIC nRF52 DK in our data collection facility, given the ease with which it can be modified. Although the modifications of the Feather have no effect on the cost of the device, adjusting the Feather requires more engineering work.

After selecting the device, our next step was to choose a suitable timing source for the power glitches. We examined two cases – the use of a cheap, readily-available and easy-to-develop product, and the use of the best available system. The Raspberry Pi 4 serves as the cheap and readily-available product, and the Active Technologies Pulse Rider PG-1072 serves as the best available system, because of its high resolution, low jitter pulse generation capabilities. Using both systems (separately), we were able to collect data traces that include the initial startup values of the devices' SRAM immediately after initializing the SRAM cells to one and performing power glitches of different durations. We then compared the quality of JULIET-PUF traces generated when using these two different setups. To perform the power glitches remotely, we connected each edge device to an S8550 transistor, with which we controlled the incoming voltage. In the Raspberry Pi's implementation, we used the Raspberry Pi's GPIO (General-Purpose Input/Output). As can be seen in Figure 5, we attached the Raspberry Pi's ground pin to the devices' ground pins, and two different outputs of the Raspberry Pi's GPIO to the transistors and to the reset pins of the devices. This allowed us to perform power glitches remotely by sending a signal to the transistors and a signal to the reset pins. To control the durations of the power glitches, we used the 'sleep' command between setting and resetting the outputs (this way we could interrupt the transistor during execution of the sleep command). Note that the signal sent to the reset pin was required when performing power glitches of short durations, since beyond some point the devices do not reboot, and thus do not send their SRAM content as required.

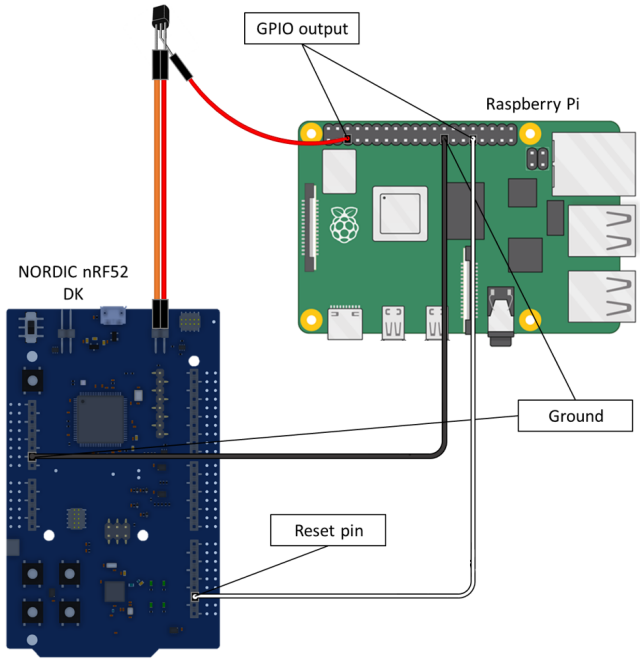

Fig. 5. Performing power glitches using the Raspberry Pi [14] [15].

Similarly, in the Pulse Rider's implementation we attached the Pulse Rider's ground pin to the devices' ground pins, and we also attached two different outputs from the Pulse Rider to the transistors and to the reset pins of the devices.

In [16], Xiao *et al.* showed that SRAM-PUFs are strongly affected by temperature variations. Therefore, for each power glitch performed, we measured the instant temperature around the devices using a Raspberry Pi Sense Hat. Figure 6 shows the data collection facility, which includes 24 devices, which in turn are connected to the power glitch source, the Raspberry Pi Sense Hat, and a management server.

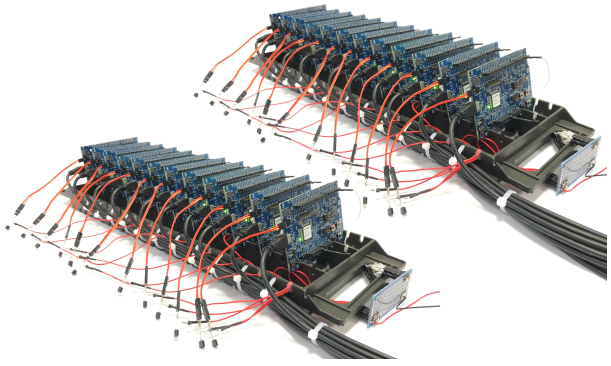

Fig. 6. JULIET-PUF's Data Collection Facility.

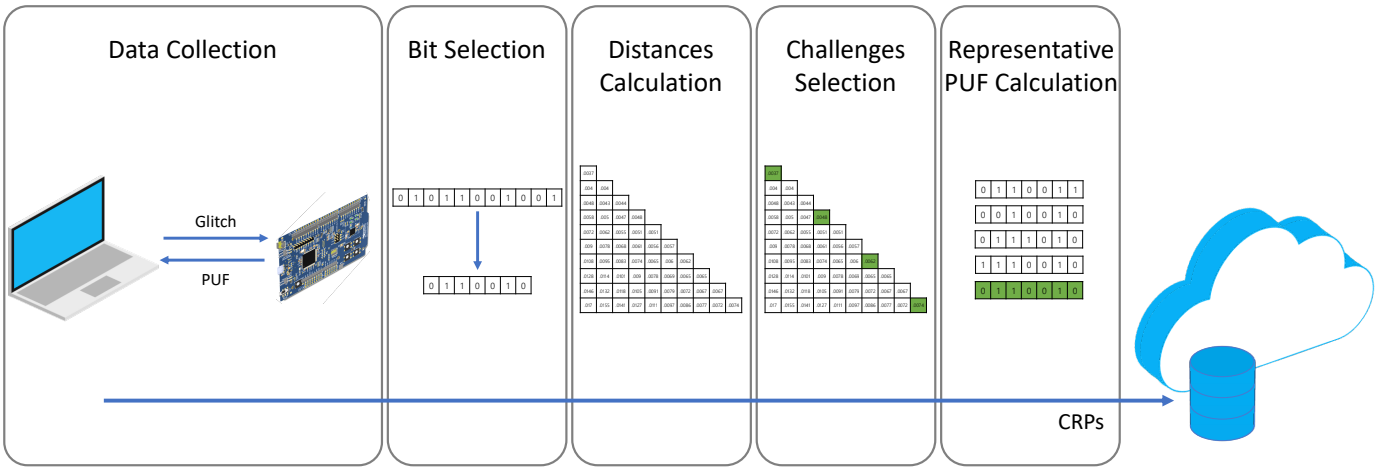

Fig. 7. The process of assembling challenge-response pairs.

B. Assembling Challenge-Response Pairs

What distinguishes JULIET-PUF from other SRAM-PUF schemes is that it has multiple CRPs. This contributes to its security: in fact, the more pairs it has, the safer it is. To this end, we collected a large amount of data and attempted to use it to extract high-quality CRPs, where the intra-distance is measurably smaller than the inter-distance. As can be seen in Figure 7, the process of assembling CRPs consists of five steps:

1) Data Collection – collecting SRAM startup values after initializing the SRAMs' cells to one and performing power glitches of varying durations.
2) Bit Selection – using bit selection approaches to identify the most appropriate bit subset for removal (meaning that without a given subset, the rest of the CRP assembly process will result in the greatest number of CRPs).
3) Distances Calculation – producing a matrix of the average distances between any two potential challenges for each device.
4) Challenges Selection – selecting the challenges that will be part of the CRPs.
5) Representative PUF Calculation – assigning the most suitable response to each challenge, by selecting each bit individually, with the value that appears most frequently in the samples.

Each step is described in more detail below.

1) Data Collection: We collected SRAM startup values from 24 IoT identical devices, after initializing the SRAMs' cells to one; them we performed power glitches of various durations, from zero to seven milliseconds, at 10 microsecond intervals (enabling the collection of 700 traces for each device). We repeated this experiment 100 times; thus, for each device we collected 100 traces for each power glitch duration, in total collecting 70,000 traces per device. Each trace consists of a timestamp, device ID, serial port, glitch duration, memory array address, device's code version, experiment tag, temperature, trigger (to indicate whether the experiment was conducted using a Raspberry Pi or Pulse Rider), and a 26KB PUF value. We first performed the process without considering the temperature around the devices during data collection. The results obtained were satisfactory, but the instability of the PUF values could be seen to reflect a cyclical behavior which was originated in the changing temperature (for example, different values were obtained at different times of the day, and a difference was seen between samples obtained on weekdays and weekends when there were fewer people in the laboratory).

2) Bit Selection: In order to obtain more accurate results, for each individual device examined, we aimed to remove the bits that were the least stable across the various glitch durations. This was accomplished by checking (for each device) each bit's stability for all samples originating from power glitches with the same duration. Then, by examining the average bit stability across all glitch durations, we were able to obtain a list of bits sorted according to their stability. We then aimed to identify the most appropriate bit subset to remove (meaning that without including a given subset, the process will result in the greatest number of CRPs). We accomplished this by repeating the entire process several times with different subsets of bits.

3) Distances Calculation: After data collection, we produced a matrix of at most 700 by 700 records for each of the 24 devices; these matrices represent the average distances between any two potential challenges (power glitch durations) for each device. The Hamming distance was calculated for each pair of samples for each of the two challenges. The average of these distances represents the average distance between each two challenges. At the end of this step, for each device we obtain a matrix whose diagonal consists of the average distances between each challenge and itself (the intra-distance), and the cells outside the diagonal consist of the average distances between each challenge and the rest of the challenges (the inter-distance). Note that the size of the matrix differs from device to device: As we exploit the SRAM remanence decay, we can exclude irrelevant regions from the matrix, ones that contain either samples identical to the initial value before the SRAM starts to decay, or a set of identical samples, having stopped changing from one to another after a certain period of time has elapsed since the device was turned off (which behave like the common use of SRAM-PUF). Those

regions can be seen as sections without a slope in Figure 10.

4) Challenges Selection: We then selected the challenges that will be part of the CRPs, using an iterative algorithm that excludes the worst challenge in each iteration, i.e., the challenge for which there is the smallest difference between the intra-distance and the smallest inter-distance. The algorithm stops when all of the remaining challenges are far from one another in a way that allows each challenge to be linked to a different response, in a one-to-one manner; therefore the remaining challenges can all be used to create the CRPs. As previously mentioned, the larger the number of CRPs, the more secure the PUF.

5) Representative PUF Calculation: To complete the CRP list assembly process, after selecting the challenges to use for each device, we assigned the most suitable response to each challenge. Since we collected 100 responses for each challenge, we assembled the representative PUF by selecting each bit individually, with the value that appears most frequently in the samples for each glitch duration.

III. Results

In this section, we describe the experiments performed and present our results. A detailed explanation of the various approaches we used to improve the results is provided, along with their impact on the security of our proposed method, which is measured by the number of the CRPs obtained. The results derived from the use of each of the two devices as a power glitch source, given all the approaches we have used are summarized in the table below (in which we use PR and Pi as abbreviations for Pulse Rider and Raspberry Pi).

	SRAM-PUF	JULIET-PUF			
		Temp control		Baseline	
		PR	Pi	PR	Pi
CRPs	1	95.58	55.91	16.54	13.54

As the table shows, introducing JULIET-PUF to an existing SRAM-PUF authentication system can enhance the system's security by a factor of between 13 and 100, depending on the temperature control and the accuracy of the power glitch source. We explain each factor in detail below.

A. Impact of the Power Glitch Source

In order to examine the accuracy of the power glitch durations that each system provides, we measured the glitch durations in practice using a Keysight MSOS604A mixed signal oscilloscope, and compared them with the intended ones. Since the data collection consists of traces obtained while performing power glitches of various durations, from zero to seven milliseconds at 10 microsecond intervals, to estimate the accuracy of each device, we measured the glitch duration we obtained in practice for various durations from 10 microseconds and up to seven milliseconds with 100 microsecond intervals.

As can be seen in Figures 8 and 9, the Pulse Rider provides much more accurate power glitches, with a significantly lower absolute error rate and standard deviation than those of the Raspberry Pi. Note that regardless of the power glitch source used, the higher the sampling rate (i.e., the smaller the interval between every two samples), the greater the number of potential CRPs, and thus the number of CRPs obtained. Also note that each device has a different number of potential CRPs and thus there is a corresponding difference in the results. This stems from the fact that the memory decay time of each device is different, as can be seen in Figure 10. We observe that if we increase the number of potential CRPs, the number of CRPs assembled at the end of the process will also increase. However, there will be a cost in doing so, as the enrollment phase will become more expensive.

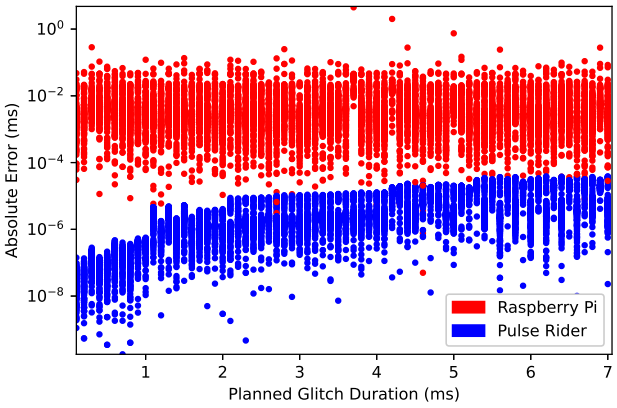

Fig. 8. Power glitch duration obtained in practice compared to the intended duration using the Pulse Rider and Raspberry Pi.

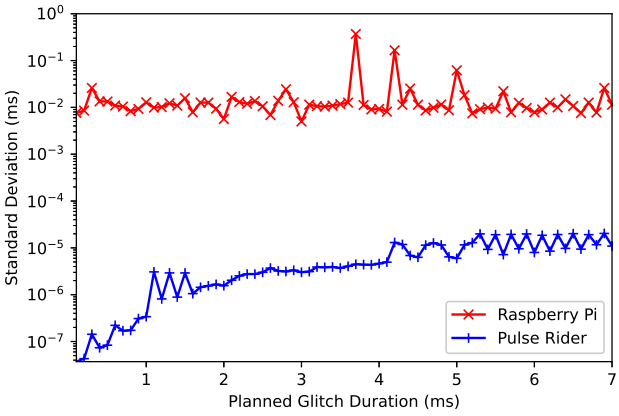

Fig. 9. The standard deviation of the power glitch duration obtained in practice using the Pulse Rider and Raspberry Pi.

The practical impact of the accuracy of the power glitch source, as observed by the number of CRPs obtained by using each source, can be seen in Table I. As shown in the Table, while both sources provided a much larger number of CRPs than the common use of SRAM-PUF, the results based on data derived from the Pulse Rider were measurably better – 16.54 times more secure (on average) than with it is with the common use of SRAM-PUF, and at least 9 times more secure in the worst case.

TABLE I
THE EFFECT OF TEMPERATURE ON THE NUMBER OF CRPS.

Device	Potential	Number of Pulse Rider CRPs			Number of Raspberry Pi CRPs		
		High Temp Cluster	Low Temp Cluster	Baseline	High Temp Cluster	Low Temp Cluster	Baseline
1	225	103	60	11	54	42	12
2	263	134	75	18	68	53	14
3	158	119	112	17	56	59	15
4	165	88	48	16	53	57	17
5	191	110	64	16	58	35	11
6	266	91	43	17	54	31	9
7	193	64	26	9	43	37	9
8	193	86	53	14	54	37	9
9	205	80	84	15	55	64	15
10	270	61	26	9	53	39	9
11	320	73	12	12	52	38	9
12	201	65	78	12	42	57	10
13	157	83	59	16	52	61	15
14	228	65	67	12	42	54	12
15	216	130	99	43	78	59	26
16	209	110	47	28	70	57	22
17	178	128	111	16	59	50	17
18	297	48	29	9	36	44	10
19	157	97	56	10	46	31	8
20	206	115	76	16	60	47	14
21	249	100	50	10	59	43	10
22	175	105	40	17	58	51	14
23	204	98	65	13	56	35	10
24	412	141	66	41	84	75	28
Average	**222.41**	**95.58**	**60.25**	**16.54**	**55.91**	**48.16**	**13.54**

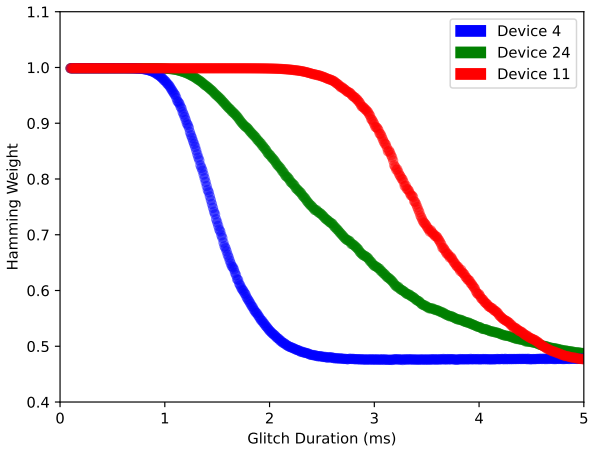

Fig. 10. Hamming weight of PUFs collected from three different units of the same IoT device, indicates that different units have different CRPs potential, influenced by their decay characteristics.

B. Impact of Temperature Change

The SRAM content of a particular device which gradually changes over the various glitch durations, is also influenced by the temperature, as can be seen in Figure 11. To examine the effect of temperature on our ability to produce CRPs, for each device we performed the process of assembling the CRPs twice more, while taking the effect of the temperature on the SRAM values into account. First, we divided the samples collected from the experiment using the Pulse Rider as the power glitch source into two groups (also referred as high temp cluster and low temp cluster): samples collected at a temperature equal to or higher than $31°C$ (as measured by the Raspberry Pi Sense Hat) and samples collected at a temperature equal to or less than $29.5°C$. Second, we divided the samples collected from the experiment using the Raspberry Pi as the glitch source into two groups: samples collected at a temperature equal to or higher than $17.75°C$ and samples collected at a temperature equal to or less than $17.25°C$. The reason for the temperature differences between the experiments is that the physical characteristics of the second experiment forced us to use a ribbon cable, to separate the Sense Hat from the heat emanating from the Raspberry Pi's CPU. In both cases, we disregarded approximately 6% of the measurements, which were obtained as the temperature was moving between the two clusters. As can also be seen in Figure 11, if we divide the samples into two clusters, the Hamming weight range of each power glitch duration can be reduced, making it possible to obtain much lower intra-distances than inter-distances. Based on this observation, for each device, we performed the process of assembling CRPs twice again (starting from step II-B3), using the data of each

cluster individually.

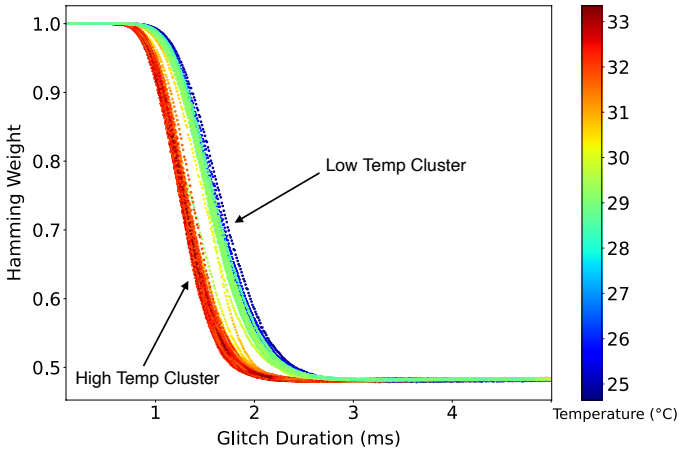

Fig. 11. The effect of temperature on the Hamming weight of samples from a particular device, with varying power glitch durations. As can be seen, if we did not control the temperature, samples with a wide range of Hamming weights would be suitable for each glitch duration. This can therefore be limited by controlling the temperature.

Table I shows that the use of the above mentioned approach improves the security of JULIET-PUF, making it 95.58 times more secure (on average) and at least 48 times more secure than the common use of SRAM-PUF, which consists of only one CRP.

C. Bit Selection

Recent studies on PUFs have demonstrated the use of helper data to avoid transmitting all of the SRAM memory as a PUF in the authentication process [17]. We examined whether the number of CRPs could be increased, by selecting a smaller subset of bits for the construction of our scheme's responses, and performing the process of assembling the CRPs accordingly. The following process applies to each device individually, as the helper data is capable of handling each device differently. First, we removed the *least stable bits* of the device, i.e., those with the highest standard deviation. After removing different percentages of these bits, we performed the process of assembling the CRPs for the device again. We found that the higher the percentage of unstable bits we removed, the fewer CRPs we obtained. Bearing in mind that the least stable bits contribute to the uniqueness of each PUF, we next tried to remove removed the *most stable bits*, i.e., those with the lowest standard deviation. Here, too, we found that bit removal reduced the number of CRPs assembled. However, when the completely stable bits were removed, i.e., those with a standard deviation equal to zero, there was no change in the number of CRPs that were assembled. As our results indicate, except for removing the completely stable bits (which do not affect the number of CRPs), the larger the broadcast budget, the more CRPs we will obtain.

D. Demo

After evaluating the CRP capacity of our system, our next step was to demonstrate that our proposed method is practical and feasible, in terms of enrollment and authentication time, for an industrial environment. The enrollment phase consists of performing the CRP assembly process described in Section II-B. In this phase, most of the time is spent on the data collection and distances calculation steps. The data collection takes about 4-5 days (based on the number of samples we collected from each power glitch). The distance calculation is a computationally intensive step which also takes several days, with the calculation of the distance matrix of each device taking between half an hour and five hours to perform, depending on the number of samples it consists of (the baseline takes all samples into account, i.e., the potential number of challenges; the clusters that are divided by temperatures, takes only samples at the desired temperatures into account). The time required to perform the other steps (i.e., bit selection, challenges selection, and representative PUF calculation) is negligible. In the demo, we used the CRPs of several devices to demonstrate the authentication phase in the following scenarios. In the first scenario, the device is a genuine device that tries to authenticate in front of the cloud server. The server chooses a challenge, and the device sends the SRAM content when it is turned back on. In this case, the authentication attempt should be successful. In the second scenario, another device tries to authenticate in front of the server, as if it were the device used in the first scenario. Because its response to the challenge is not the same as that of the first device, the authentication fails. In the third, adversarial scenario, we show that JULIET-PUF is more resistant to eavesdropping attacks than a standard SRAM-PUF. In this scenario, another device, which represents a counterfeiting adversary, eavesdrops on the authentication attempt of the first device, stores the response in its own memory and finally sends the response when it is required to authenticate itself. Because the server chooses a different challenge, the authentication fails. In all three scenarios, the devices were approved or rejected within seven seconds. Note that in the authentication phase, most of the time is spent on broadcasting the SRAM content, which means that using a faster interface or transmitting less data will reduce the time. A brief demo video is provided in the link: https://drive.google.com/file/d/1LFYXt40qkg1dw4kSv7EKkg1dJmy7K-2O/view?usp=sharing.

IV. DISCUSSION

The results of our evaluation demonstrate that we were able to provide a sustainable system that significantly improves the security of SRAM-PUF, without any additional hardware costs. We also conclude that there are a number of factors with which the results can be further improved:

1) Power glitch duration accuracy – We aimed to create an affordable solution that would improve SRAM-PUF security without the need for additional hardware. Therefore, we used a Raspberry Pi to generate the power glitches. To

better understand the limitations of this solution, we also used a Pulse Rider. As can be seen from the results, with a higher ability to produce accurate power glitches, it is possible to produce more CRPs. It would be interesting to consider a solution that involves producing more accurate power glitches while maintaining a low overall system cost.

2) Temperature accuracy – Our experiments demonstrated the impact of temperature on the number of CRPs. While we examined dividing the samples into two groups, dividing the samples into more than two groups would improve the accuracy of the results and thus lead to more CRPs.
3) Hardening the challenge – It is possible to significantly increase the number of CRPs by creating a more complex challenge, e.g., by querying the end device more than once or initializing the SRAM values to values other than one before the power glitch is performed.

In the subsections that follow, we discuss the cost and reliability impact of our proposed method, provide a brief security analysis of it, introduce prior work on SRAM-PUF design and evaluation, and conclude the paper.

A. Impact on Cost and Reliability

Although JULIET-PUFdoes not require any additional hardware costs on the edge device, this challenge-response scheme has additional steps, compared to traditional SRAM-based challenge-response schemes. This is reflected in the impact on the cost, and possibly also on the lifetime of the devices. For example, the common use of SRAM-PUF does not require a timekeeping mechanism, and does not involve power glitches on the edge devices. Hence, the enrollment process is shorter and simpler, and there is no concern about the impact of the scheme on the lifetime of the devices. Below we outline the extra cost involved in using the scheme, and our assessment of its impact on the devices.

Impact on Cost:

1) Hardware cost and engineering time: As mentioned in the methods section, turning the devices off and on for varying periods of time requires the use of transistors (one for each device) and connecting wires. In addition to the per-device change, the manufacturing facility must also install a device that will serve as the power glitch source. We proposed the Raspberry Pi 4 as the cheap and readily-available product, and the Active Technologies Pulse Rider PG-1072 as the best available system. Modifying the device hardware will probably incur additional engineering time to design the change and test it, costing an extra several days of engineering work.
2) Enrollment time: in addition to the small change to the device's bill of materials, the devices also take longer to leave the factory after they produced. As explained in more detail in III-D, most of the time of the enrollment phase is spent on the data collection and distances calculation steps. The data collection can take several days, based on the number of samples collected from each power glitch. The distance calculation is a computationally intensive step which also takes several days, with the calculation of the distance matrix of each device taking between half an hour and five hours to perform, depending on the number of samples. In this work we did not take into account the efficiency of the implementation, and we estimate that the enrollment time can be shortened significantly. In addition, the time that this phase requires is not influenced by the number of the devices. Hence, it can be done on a large batch of devices in parallel, resulting in a very low amortized time per device.

Impact on Lifetime: During the enrollment phase, the device is turned on and off approximately 1,000 times. After concluding this step, however, the device is only powered down a single time for the purpose of authentication. Due to the nature of the IoT-based deployment, we assume authentication will only be performed a few times a month or a year, depending on its implementation. Note that we do not provide an invalid input voltage at any point, but only turn off the device. This should have no noteworthy effect on the devices' lifetime or the number of cycles before failure, according to an official response from the manufacturer of the development kit [18]. The factors that do affect the device lifetime are mainly the supplied voltage, the storage and operating temperature, and the amount of flash write/erase cycles [19]. In general, the number of power cycles performed during the JULIET-PUF enrollment phase is much lower than the number of power cycles considered in modern works discussing power cycle reliability. [20] [21] [19].

To conclude, while the JULIET-PUF challenge-response scheme does have slightly higher costs than the direct SRAM-PUF challenge-response scheme, we consider this cost is more than offset by the increased security level that it provides.

B. Security Analysis

JULIET-PUF's security should be considered in the context of the IoT counterfeiting threat model. Under this threat model, the adversary can purchase an arbitrary number of edge devices and reverse-engineer them in the lab to discover their secrets. Then, the adversary can try to use the extracted secrets to create a counterfeit device that behaves the same way as the original device. A unique aspect of the counterfeiting setting is the matter of cost – if the cost of manufacturing a counterfeit device exceeds the cost of purchasing a non-counterfeit device (the original device), then the attack is considered ineffective.

The analysis of JULIET-PUF's security under various attacks presented below, shows that it is equal or superior to that of standard SRAM-PUF-based authentication. Table II summarizes the findings of our security analysis.

Man-in-the Middle and Replay Attacks: In this attack setting, the adversary monitors the communication between the authenticator and a legitimate edge device, and records the responses sent by the device. Next, the adversary creates a counterfeit device which is identical to the legitimate device, but has malicious firmware which always replays the recorded response whenever it is queried. This attack vector is already

made impossible in the SRAM-PUF setting, as long as industry standard transport layer encryption and cryptographic key exchange protocols are followed.

Malicious Firmware: In this attack setting, the adversary loads malicious firmware to a legitimate edge device. The malicious firmware causes the device to output its PUF response without any encryption. Next, the adversary creates a counterfeit edge device which is identical to the legitimate device, but has malicious firmware which contains an embedded copy of the extracted PUF response. A local attacker with a flash firmware programmer could understand the SRAM-PUF implementation. While the firmware can be copy-protected using traditional methods such as setting read lock fuse bits, invasive and non-invasive attacks have been shown to bypass such protection [22]. When the counterfeit edge device powers up, it overwrites its own PUF response with the extracted PUF response, and then follows any required cryptographic protocol as if the legitimate device's extracted PUF response was its own. Standard SRAM-PUF protection is completely defeated by this attack, since there is only a single possible PUF response, and the malicious firmware puts it in the hands of the adversary. JULIET-PUF, on the other hand, remains safe, since the counterfeit device has no way of measuring the power glitch duration. As a result, the counterfeit device does not know which possible response to send to the authenticator, resulting in a very low authentication success rate.

Malicious Firmware and Modified Hardware: This attack setting extends the previous setting, in which the adversary extracts the PUF response from a captive legitimate device. In this setting, the counterfeit device is not completely identical to the edge device; instead, it contains specific hardware designed to defeat JULIET-PUF. Such counterfeit device should minimally contain a sensitive timer, a temperature sensor, a large amount of non-volatile memory to store the extracted PUF responses, and, finally, a battery backup which enables the sensors to work even when the device is turned off. This is a considerable modification of the original hardware, which immediately renders the OEM overproduction use case irrelevant, since the hardware used for the original edge device cannot be used in this attack setting. This fact, along with the costs associated with the additional hardware and developing and testing the counterfeit device, make this approach financially impractical for the IoT edge device use case. While, in principle, an adversary can use advanced invasive hardware modification techniques such as focused ion beam (FIB) circuit editing to modify an existing device's JULIET-PUF response to match the response of another device, the cost of doing so is prohibitively expensive, thus making it even more irrelevant to the counterfeiting use case [23].

DDoS Attacks: In this attack setting, the adversary makes repeated attempts to authenticate against the cloud server. Although the adversary is unable to perform a successful authentication, because it does not possess the correct response to the challenge sent by the server, given an effective DDOS attack, it can make the authentication service unavailable. By doing so, the adversary will effectively prevent authentic devices from being able to connect to the server for the duration of the attack. This form of attack affects our system as well as any other server-based system.

Brute Force Attacks: In this attack setting, the adversary makes repeated attempts to authenticate against cloud server, in a trial and error fashion, with the aim of eventually sending the correct response to the challenge sent by the server. In relation to standard brute force scenarios, JULIET-PUFmakes it particularly difficult for the adversary, since the challenge changes with each authentication attempt, increasing the space of responses which must be enumerated. However, we estimate that if the adversary knows how the scheme works, the range of possible responses can be reduced.

Modeling Attacks: In this attack setting, the adversary has a limited set of CRPs and tries to create a model that predicts the PUF response to a given challenge [24]. JULIET-PUFcan be vulnerable to such an attack if not crafted with caution. In our scheme, the challenge is the duration of the power glitch performed on the device. If an adversary can obtain a raw CRP response with a short glitch duration, this response can be used to predict the response in the case of a longer glitch. For example, lets examine the case where the SRAM is initialized to logical 1. The adversary has possession of a single CRP, i.e. glitch duration of time t (the challenge) denoted as C_t, and a response R which is the values of the SRAM after performing the challenge. We denote R_i as the i'th bit in the response. If $R_i = 0$ after performing C_t, there are 2 options in such case:

(a) R_i is random
(b) R_i is 0-skewed

In case a the adversary can learn nothing about the value of R_i in other challenges. On the other hand, case b means R_i needs roughly t seconds glitch to reach its stable state 0, thus, for any C'_t where $t' > t$ the adversary can predict $R_i = 0$. To overcome such an attack, the response should be processed before it is sent (for example by hashing or encryption) to prevent its raw contents from being accessed by an adversary. Such a hashing scheme should be carefully chosen, considering the low signal to noise ratio of the recovered PUF signal.

C. Related Work

Many studies have focused on SRAM-PUF's design and evaluation, with the aim of improving its characteristics, particularly its reliability and its unpredictability.

Physically unclonable functions were initially introduced in 2002 by Pappu *et al.* [25] as physical one-way functions that can be used to allocate and authenticate unique identifiers by converting physical variations into fixed-length strings of binary digits. In [26], Gassend *et al.* extended the PUF domain by introducing silicon physical random functions, which are designed to identify and authenticate integrated circuits (ICs) using manufacturing variations across ICs. Relying on the statistical delay variations of wires and devices on different ICs, they created a parameterized self-oscillating circuit for the characterization of ICs. In recent years, this implementation has been referred to as a Ring Oscillator PUF. In addition to the development of intrinsic PUFs based on delay measurements, a method in which memory cells' startup values are measured and utilized for digital fingerprinting was developed.

TABLE II
SECURITY ANALYSIS

Threats	Standard SRAM-PUF	JULIET-PUF
Eavesdropping and Replay	Secure	Secure
Malicious Firmware	Insecure	Secure
Malicious Firmware, Modified Hardware	Insecure	Insecure (impractical cost)
DDOS	Insecure	Insecure
Brute Force	Insecure	Secure
PUF Modeling	Insecure	Insecure

This method is now known as SRAM-PUF, and it is widely used [27], [28].

Several studies focused on designing an SRAM-PUF with improved characteristics. In [29], Garg *et al.* introduced a technique for improving the reliability of SRAM-PUF, utilizing its aging effects. By controlling the polarity of aging in SRAM arrays, the proposed technique maintains the uniformity of SRAM-PUF, i.e., an even distribution of ones and zeros. After achieving the target uniformity, they improved the reliability by aging the SRAM arrays further. In [30], Aung *et al.* analyzed SRAM-PUF characteristics found in IoT microcontrollers using temporal majority voting (TMV) and data remanence methods. Their analysis covered SRAM-PUF's biasness, uniqueness, and stability, which they examined on different microcontrollers. Using these methods, they were able to obtain an SRAM-PUF of 128 bits out of 512, with an error rate of 3.77×10^{-8} and stability of 99.983%, results that make it suitable for simple microcontrollers. In [31], Böhm *et al.* implemented an SRAM-PUF on microcontrollers and by using repetition correction code, they reduced the error rate to $6.85 \times 10^{-7}\%$. In [16], Xiao *et al.* used bit analysis and bit selection algorithms to reduce the high test time and design overheads of SRAM-PUFs, which have made them unsuitable for high volume production. In the process, they pointed out conditions in which stable SRAM cells can be selected for PUFs. In [32], Xu *et al.* proposed a DRV-based hash function which is insensitive to temperature and enables the implementation of a PUF that utilizes the variation sensitivity of SRAM data retention voltage (DRV), the minimum voltage required for a cell to maintain its state. In [33], Xu *et al.* showed that when instances of identical storage cells are compared with respect to failure propensity, it is possible to create a high quality PUF. In this context, they proposed a failure-based PUF which uses failures caused by control of power gating durations.

Other studies have focused on investigating the suitability of different SRAMs for use as PUF primitives. In [1], Barbareschi *et al.* investigated the suitability of various types of 90nm SRAM devices based on PUFs' quality parameters: reliability, uniqueness, and uniformity. This was done by studying the behavior of the SRAMs' startup patterns under different power supply strategies. In [34], Schrijen *et al.* investigated different SRAM memories, which were used as PUF primitives, in SRAM design on technologies varying from 65nm to 180nm, based on PUFs' quality parameters, emphasizing reliability and uniqueness. Using the startup patterns of different SRAMs, measured under various conditions, such as temperature and applied voltage, the authors were able to show that all of the examined SRAMs are suitable for use as PUFs.

Although SRAM-PUF is a great solution for authenticating resource-constrained devices with limited computation power and memory, relying on just SRAM is often insufficient, because SRAM is considered a weak-PUF. As stated by Guajardo *et al.* in [27], a weak-PUF is a PUF which has a small number of CRPs, while a strong-PUF has a large number of available CRPs, so large that the likelihood of a time-limited attack based on exhaustively measuring the CRPs to succeed is negligible. Transforming SRAM-PUF into a strong PUF is not an easy task. In [35], Farha *et al.* proposed an SRAM-PUF-based authentication scheme suitable for low-resource IoT edge devices; the proposed scheme uses re-ordered memory addresses as challenges and the corresponding SRAM cells' startup values as responses. In a comment paper on this work [36], Amar *et al.* analyzed this scheme and showed that while it claims to offer strong PUF functionality, the scheme creates a weak PUF. In fact, an active attacker can read out the device's entire PUF response after a very small number of queries are exchanged with the prover.

Several studies have focused on ways of overcoming PUFs by predicting the response of the device. For example, in [7], Cortez *et al.* suggested a method for the analysis and prediction of the SRAM signature based on variables such as the transistor length, material thickness, temperature, and voltage. To do so, the authors calculated the static noise margin (SNM) value and analyzed its variability in changing environments. In [12], Zeitouni *et al.* introduced a side-channel attack, which utilizes the remanence decay in volatile memory, and demonstrated how a non-invasive clone attack can be launched against SRAM-PUFs. They also showed that this attack is feasible against small memory-based PUFs, even without the use of specialized lab equipment.

Although SRAM is commonly used as a PUF due to its simplicity and availability in many electronic devices, it has a major drawback - it decays and becomes less consistent over time. Some physical phenomena in a silicon integrated circuit cause the circuit's parameters to slowly change and thus change the PUF. Maes *et al.* [37] studied the reliability of SRAM-PUF in light of the data-dependent silicon aging. The authors also proposed anti-aging techniques for SRAM-PUFs which are based solely on data-dependent silicon aging effects observed in regular SRAM cells over the integrated circuit's lifetime; the proposed techniques can be used on any

standard SRAM without incurring pre-deployment overhead.

D. Conclusion

In the IoT domain, new security issues are emerging, while traditional security issues are becoming more difficult. Therefore, the need for entity authentication of end devices, which is considered an essential aspect of IoT system security today, is growing. Because traditional ID mechanisms are infeasible in IoT devices due to the constrained runtime environment of the edge devices and the additional costs and deployment issues they introduce, alternative solutions for securing IoT components are required. In light of this, we propose JULIET-PUF, a novel PUF-based unique ID generation method that relies on SRAM content retrieval after power glitches of various durations. Our evaluation on a dataset of traces from multiple units of a popular commercial off-the-shelf IoT device shows that JULIET-PUF offers a considerable security advantage over standard SRAM-PUF in the counterfeiting threat model, all without requiring any additional hardware costs. Future work may focus on adapting JULIET-PUF for use in other domains, such as cellular devices and computer components.

Acknowledgements

This research was funded by the Alibaba DAMO Academy.

REFERENCES

[1] M. Barbareschi, E. Battista, A. Mazzeo, and N. Mazzocca, "Testing 90 nm microcontroller SRAM PUF quality," in *10th International Conference on Design & Technology of Integrated Systems in Nanoscale Era, DTIS 2015, Napoli, Italy, April 21-23, 2015*. IEEE, 2015, pp. 1–6. [Online]. Available: https://doi.org/10.1109/DTIS.2015.7127360

[2] P. H. Nguyen, D. P. Sahoo, C. Jin, K. Mahmood, U. Rührmair, and M. van Dijk, "The interpose PUF: secure PUF design against state-of-the-art machine learning attacks," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2019, no. 4, pp. 243–290, 2019. [Online]. Available: https://doi.org/10.13154/tches.v2019.i4.243-290

[3] A. R. Korenda, S. Assiri, F. Afghah, and B. Cambou, "An error correction approach to memristors puf-based key encapsulation," in *2021 IEEE International Conference on Omni-Layer Intelligent Systems, COINS 2021, Barcelona, Spain, August 23-25, 2021*. IEEE, 2021, pp. 1–6. [Online]. Available: https://doi.org/10.1109/COINS51742.2021.9524282

[4] S. Mueelich and M. Bossert, "A new error correction scheme for physical unclonable functions," in *SCC 2017; 11th International ITG Conference on Systems, Communications and Coding*, 2017, pp. 1–6.

[5] M. kalya and S. Kumar, "Low complexity ldpc error correction code for modified anderson puf to improve its uniformity," in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, 2020, pp. 997–1002.

[6] W. Xiong, A. Schaller, S. Katzenbeisser, and J. Szefer, "Dynamic physically unclonable functions," in *Proceedings of the 2019 on Great Lakes Symposium on VLSI, GLSVLSI 2019, Tysons Corner, VA, USA, May 9-11, 2019*, H. Homayoun, B. Taskin, T. Mohsenin, and W. Zhao, Eds. ACM, 2019, pp. 311–314. [Online]. Available: https://doi.org/10.1145/3299874.3318025

[7] M. Cortez, A. Dargar, S. Hamdioui, and G. J. Schrijen, "Modeling SRAM start-up behavior for physical unclonable functions," in *2012 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, DFT 2012, Austin, TX, USA, October 3-5, 2012*. IEEE Computer Society, 2012, pp. 1–6. [Online]. Available: https://doi.org/10.1109/DFT.2012.6378190

[8] M. S. Mispan, S. Duan, B. Halak, and M. Zwolinski, "A reliable PUF in a dual function SRAM," *Integr.*, vol. 68, pp. 12–21, 2019. [Online]. Available: https://doi.org/10.1016/j.vlsi.2019.06.001

[9] R. Cramer, I. Damgård, and P. D. MacKenzie, "Efficient zero-knowledge proofs of knowledge without intractability assumptions," in *Public Key Cryptography, Third International Workshop on Practice and Theory in Public Key Cryptography, PKC 2000, Melbourne, Victoria, Australia, January 18-20, 2000, Proceedings*, ser. Lecture Notes in Computer Science, H. Imai and Y. Zheng, Eds., vol. 1751. Springer, 2000, pp. 354–373. [Online]. Available: https://doi.org/10.1007/978-3-540-46588-1_24

[10] S. Almuhammadi and C. Neuman, "Security and privacy using one-round zero-knowledge proofs," in *7th IEEE International Conference on E-Commerce Technology (CEC 2005), 19-22 July 2005, München, Germany*. IEEE Computer Society, 2005, pp. 435–438. [Online]. Available: https://doi.org/10.1109/ICECT.2005.78

[11] A. Rahmati, M. Salajegheh, D. E. Holcomb, J. Sorber, W. P. Burleson, and K. Fu, "TARDIS: time and remanence decay in SRAM to implement secure protocols on embedded devices without clocks," in *Proceedings of the 21th USENIX Security Symposium, Bellevue, WA, USA, August 8-10, 2012*, T. Kohno, Ed. USENIX Association, 2012, pp. 221–236. [Online]. Available: https://www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/rahmati

[12] S. Zeitouni, Y. Oren, C. Wachsmann, P. Koeberl, and A. Sadeghi, "Remanence decay side-channel: The PUF case," *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 6, pp. 1106–1116, 2016. [Online]. Available: https://doi.org/10.1109/TIFS.2015.2512534

[13] "Nordic semiconductor," https://www.nordicsemi.com/-/media/Software-and-other-downloads/Product-Briefs/nRF52832-product-brief.pdf.

[14] "raspberrypi.com," https://www.raspberrypi.com/documentation/computers/images/GPIO-Pinout-Diagram-2.png.

[15] "Nordic semiconductor," https://www.nordicsemi.com/Products/Development-hardware/nrf52-dk.

[16] K. Xiao, M. T. Rahman, D. Forte, Y. Huang, M. Su, and M. Tehranipoor, "Bit selection algorithm suitable for high-volume production of SRAM-PUF," in *2014 IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2014, Arlington, VA, USA, May 6-7, 2014*. IEEE Computer Society, 2014, pp. 101–106. [Online]. Available: https://doi.org/10.1109/HST.2014.6855578

[17] R. F. H. Fischer and S. Müelich, "A new helper data scheme for soft-decision decoding of binary physical unclonable functions," *IEEE Access*, vol. 10, pp. 12 644–12 653, 2022. [Online]. Available: https://doi.org/10.1109/ACCESS.2022.3146989

[18] S. Rømo, "Power cycling effect on mean time to failure," https://devzone.nordicsemi.com/f/nordic-q-a/95316/power-cycling-effect-on-mean-time-to-failure.

[19] L. R. GopiReddy, L. M. Tolbert, and B. Ozpineci, "Power cycle testing of power switches: A literature survey," *IEEE Transactions on Power Electronics*, vol. 30, no. 5, pp. 2465–2473, 2014.

[20] U.-M. Choi, F. Blaabjerg, and S. Jørgensen, "Power cycling test methods for reliability assessment of power device modules in respect to temperature stress," *IEEE Transactions on Power Electronics*, vol. 33, no. 3, pp. 2531–2551, 2017.

[21] U.-M. Choi, S. Jørgensen, and F. Blaabjerg, "Advanced accelerated power cycling test for reliability investigation of power device modules," *IEEE Transactions on Power Electronics*, vol. 31, no. 12, pp. 8371–8386, 2016.

[22] S. Skorobogatov, "Copy protection in modern microcontrollers," https://www.cl.cam.ac.uk/~sps32/mcu_lock.html.

[23] C. Helfmeier, C. Boit, D. Nedospasov, and J. Seifert, "Cloning physically unclonable functions," in *HOST*. IEEE Computer Society, 2013, pp. 1–6.

[24] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010*, E. Al-Shaer, A. D. Keromytis, and V. Shmatikov, Eds. ACM, 2010, pp. 237–249. [Online]. Available: https://doi.org/10.1145/1866307.1866335

[25] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Science*, vol. 297, no. 5589, pp. 2026–2030, 2002. [Online]. Available: https://science.sciencemag.org/content/297/5589/2026

[26] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, ser. CCS '02. New York, NY, USA: Association for Computing Machinery, 2002, pp. 148–160. [Online]. Available: https://doi.org/10.1145/586110.586132

[27] J. Guajardo, S. S. Kumar, G. J. Schrijen, and P. Tuyls, "FPGA intrinsic pufs and their use for IP protection," in *Cryptographic*

Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings, ser. Lecture Notes in Computer Science, P. Paillier and I. Verbauwhede, Eds., vol. 4727. Springer, 2007, pp. 63–80. [Online]. Available: https://doi.org/10.1007/978-3-540-74735-2_5

[28] D. E. Holcomb, W. P. Burleson, K. Fu *et al.*, "Initial sram state as a fingerprint and source of true random numbers for rfid tags," in *Proceedings of the Conference on RFID Security*, vol. 7, 2007, p. 01.

[29] A. Garg and T. T. Kim, "Design of SRAM PUF with improved uniformity and reliability utilizing device aging effect," in *IEEE International Symposium on Circuits and Systemss, ISCAS 2014, Melbourne, Victoria, Australia, June 1-5, 2014*. IEEE, 2014, pp. 1941–1944. [Online]. Available: https://doi.org/10.1109/ISCAS.2014.6865541

[30] P. P. Aung, K. Mashiko, N. B. Ismail, and C. Y. Ooi, "Evaluation of SRAM PUF characteristics and generation of stable bits for iot security," in *Emerging Trends in Intelligent Computing and Informatics - Data Science, Intelligent Information Systems and Smart Computing, International Conference of Reliable Information and Communication Technology, IRICT 2019, Johor, Malaysia, 22-23 September, 2019*, ser. Advances in Intelligent Systems and Computing, F. Saeed, F. Mohammed, and N. Gazem, Eds., vol. 1073. Springer, 2019, pp. 441–450. [Online]. Available: https://doi.org/10.1007/978-3-030-33582-3_42

[31] C. Böhm, M. Hofer, and W. Pribyl, "A microcontroller SRAM-PUF," in *5th International Conference on Network and System Security, NSS 2011, Milan, Italy, September 6-8, 2011*, P. Samarati, S. Foresti, J. Hu, and G. Livraga, Eds. IEEE, 2011, pp. 269–273. [Online]. Available: https://doi.org/10.1109/ICNSS.2011.6060013

[32] X. Xu, A. Rahmati, D. E. Holcomb, K. Fu, and W. P. Burleson, "Reliable physical unclonable functions using data retention voltage of SRAM cells," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 34, no. 6, pp. 903–914, 2015. [Online]. Available: https://doi.org/10.1109/TCAD.2015.2418288

[33] X. Xu and D. E. Holcomb, "Reliable PUF design using failure patterns from time-controlled power gating," in *2016 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, DFT 2016, Storrs, CT, USA, September 19-20, 2016*. IEEE Computer Society, 2016, pp. 135–140. [Online]. Available: https://doi.org/10.1109/DFT.2016.7684085

[34] G. J. Schrijen and V. van der Leest, "Comparative analysis of SRAM memories used as PUF primitives," in *2012 Design, Automation & Test in Europe Conference & Exhibition, DATE 2012, Dresden, Germany, March 12-16, 2012*, W. Rosenstiel and L. Thiele, Eds. IEEE, 2012, pp. 1319–1324. [Online]. Available: https://doi.org/10.1109/DATE.2012.6176696

[35] F. Farha, H. Ning, K. Ali, L. Chen, and C. Nugent, "Sram-puf-based entities authentication scheme for resource-constrained iot devices," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5904–5913, 2021. [Online]. Available: https://doi.org/10.1109/JIOT.2020.3032518

[36] M. Amar, A. Kama, K. Wang, and Y. Oren, "Comment on "sram-puf based entities authentication scheme for resource-constrained iot devices"," *Manuscript submitted for publication*, 2021.

[37] R. Maes and V. van der Leest, "Countering the effects of silicon aging on SRAM pufs," in *2014 IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2014, Arlington, VA, USA, May 6-7, 2014*. IEEE Computer Society, 2014, pp. 148–153. [Online]. Available: https://doi.org/10.1109/HST.2014.6855586

Amit Kama is an M.Sc. student in the cyber security track of the Department of Software and Information Systems Engineering at Ben-Gurion University of the Negev (BGU), Israel. He received his B.Sc. in computer science from the Open University of Israel and is currently a member of BGU's Implementation Security and Side-Channel Attacks Lab. His current research interests include physically unclonable functions (PUFs), side-channel analysis and machine learning.

Michael Amar is pursuing his M.Sc. in the Department of Software and Information Systems Engineering at BGU and is currently a member of BGU's Implementation Security and Side-Channel Attacks Lab. His research interests are side-channel analysis and machine learning.

Snir Gaaton is pursuing his B.Sc. in information systems engineering in the Department of Software and Information Systems Engineering at BGU and is currently a member of BGU's Implementation Security and Side-Channel Attacks Lab.

Kang Wang is a staff security engineer on the security research team of the Alibaba Group. He received his B.Sc. from Tsinghua University. He was a speaker at Black Hat Europe 2015, Black Hat USA 2017/2018, Virus Bulletin 2018, HITB Dubai 2018, and Black Hat Asia 2019.

Yifan Tu is a senior staff engineer of ele.me, the online on-demand food delivery service of Alibaba Group. He is responsible for the risk management technology development at ele.me. Prior to that, he designed a DDoS mitigation system in Alibaba Cloud.

Yossi Oren (SM'17) received his M.Sc. degree in computer science from the Weizmann Institute of Science, Rehovot, Israel and Ph.D. degree in electrical engineering from Tel Aviv University, Tel Aviv, Israel. He is a senior lecturer (assistant professor) in BGU's Department of Software and Information Systems Engineering, where he studies implementation security and side-channel attacks.