Research

A novel IoT sensor authentication using HaLo extraction method and memory chip variability

Holden Gordon[1] · Thomas Lyp[1] · Calvin Kimbro[1] · Sara Tehranipoor[1]

Received: 23 August 2021 / Accepted: 24 September 2021
Published online: 20 October 2021

Abstract
Since the inception of encrypted messages thousands of years ago, mathematicians and scientists have continued to improve encryption algorithms in order to create more secure means of communication. These improvements came by means of more complex encryption algorithms that have stronger security features such as larger keys and trusted third parties. While many new processors can handle these more complex encryption algorithms, IoT devices on the edge often struggle to keep up with resource intensive encryption standards. In order to meet this demand for lightweight, secure encryption on the edge, this paper proposes a novel solution, called the High and Low (HaLo) method, that generates Physical Unclonable Function (PUF) signatures based on process variations within flash memory. These PUF signatures can be used to uniquely identify and authenticate remote sensors, and help ensure that messages being sent from remote sensors are encrypted adequately without requiring computationally expensive methods. The HaLo method consumes 20x less power than conventional authentication schemes commonly used with IoT devices, it has an average latency of only 39ms for 512 bit signature generation, and the average error rate is below 0.06%. Due to its low latency, low error rate, and high power efficiency, the HaLo method can progress the field of IoT encryption standards by accurately and efficiently authenticating remote sensors without sacrificing encryption integrity.

Keywords Physical Unclonable Function (PUF) · Internet of Things (IoT) · Flash memory

Mathematics Subject Classification MSC code1 · MSC code2 · more

1 Introduction

Internet of Things (IoT) sensors have seen explosive growth over the last twenty years. The consumer IoT market is estimated to reach 142 billion dollars by 2026 at a compound annual growth rate of 17%. Estimates forecast that by 2025, there will be 152,200 IoT devices connecting to the internet every single minute. While the expanding IoT landscape has created a space to collect vast amounts of data and help automate simple and dangerous tasks, it has also increased the potential for attacks. Malware attacks targeting IoT devices have increased by 30%, and 76% of IoT risk professionals believe that their organization's IoT security posture leaves them vulnerable. This is particularly important for the IoT devices used as medical sensors in telehealth applications.

Telecommunication in healthcare, or telehealth, provides a means by which patients can interact with medical professionals and health-related services virtually. In light of the COVID-19 worldwide pandemic, the need for secure and readily

✉ Sara Tehranipoor, ftehranipoor@scu.edu; Holden Gordon, hgordon@alumni.scu.edu; Thomas Lyp, tlyp@scu.edu; Calvin Kimbro, ckimbro@scu.edu | [1]Santa Clara University, 500 El Camino Real, Santa Clara, CA 95053, USA.

available remote patient monitoring has never been more evident. Rural and low-income communities, in particular, have been severely impacted by the lack of accessibility to in-person healthcare. This has created the need for access to remote patient monitoring and virtual health visits to provide accessibility for premier care for all patients. However, the convenience of connecting medical providers with patients remotely also introduces significant security and privacy risks. One such risk of using unsecured medical devices is the potential for a major privacy breach, as they store sensitive information such as vital signals, diagnosed conditions, therapies, and a variety of personal data [1]. On the dark web, an individual's private health information (PHI) can sell up to 100 times the value of a social security number or credit card. This creates a strong financial incentive for malicious actors to steal healthcare data from small embedded telehealth sensors which are vulnerable due to tight processing, energy, and latency requirements of IoT devices.

Physical Unclonable Functions (PUFs) can provide a lightweight and tamper-proof security primitive for IoT devices particularly telehealth sensors. PUFs create cryptographic signatures that can be used for authentication, software attestation, and cryptographic key generation. These signatures are derived from the sub-micron process variations present in integrated circuits (ICs). These signatures are never stored on the device itself and in many cases are extremely difficult to tamper. Specific applications of medical IoT telehealth devices have been investigated. For example work from Palve et al. [2] investigates networking security solutions for edge IoT devices. However, it does not consider the advantages that PUF technology can provide. Secondly, Farkoon et al. [3] also suggest that PUFs can provide unique advantages for telehealth sensor security. However, it is not implemented in physical hardware; therefore, performance constraints, financial considerations, and aging considerations cannot be fully considered.

Therefore, this paper proposes a novel Flash PUF that solely utilizes commercial and off the shelf components leveraging a unique PUF extraction scheme known as the High and Low Method (HaLo) method. It enhances previous Flash PUF implementations by providing both high accuracy, entropy, and near ideal inter-Hamming distance without modifying off the shelf flash chips. Secondly, The HaLo method aims at minimizing authentication latency and maximizing the lifetime of the flash chip by limiting program/erase cycles (PECs) while avoiding costly peripheral security chips. Furthermore, the entire development environment costs only $15 dollars allowing for further research and affordable edge deployment in a commercial setting.

Secondly, an authentication handshaking mechanism is also used to connect the PUF enabled health sensor to a Raspberry Pi gateway and the authentication's latency and power is profiled highlighting a clear performance advantage to using the HaLo enabled Flash PUF. Overall, our main contributions are summarized as follows:

- We have built a novel PUF extraction technique called the HaLo method, which supports edge deployment on low-cost microcontrollers. This will lower the cost of entry, and help encourage secure data transmission for remote devices.
- The proposed HaLo method offers a PUF solution for accurate authentication and has lower latency and lower power consumption than other PUF generation techniques and legacy pre-shared key security systems.
- The HaLo method is compatible with off-the-shelf ONFI 2.2 compliant flash chips, which could lead to backward compatibility implementation on existing health sensors.

The organization of the remainder of this paper is as follows. First, Sect. 2 will discuss the preliminary background information required in order to understand the HaLo method. Next, Sect. 3 will give an in-depth look at related published authentication schemes using process variation, as well as a look at the advantages of the novel HaLo extraction method. Section 4 will discuss the HaLo extraction method in detail including design constraints and considerations, and Sect. 5 will provide experimental results and validation of the novel HaLo method. Finally, Sect. 6 will provide details on the telehealth application proof of concept built with the HaLo extraction method, and Sect. 7 will give a brief conclusion and summary of the work, along with details on proposed future work for this project.

2 Preliminaries

In this section we will briefly discuss the general background information involving our proposed authentication scheme. Specifically, this section provides an overview of current wireless authentication protocol solutions, it gives a brief introduction to process variations found in flash memory chips, and it will provide a basic understanding of PUFs.

2.1 Wireless authentication methods

There are a variety of wireless authentication methods that are used by IoT and Telehealth devices. One of the most common ways to authenticate these resource constrained devices is by utilizing pre-shared keys [4]. Pre-shared keys are authentication keys or tokens that are shared with a device prior to its deployment. These keys are then exchanged with a gateway in order to authenticate an IoT device. There are several important vulnerabilities within this model that are solution wishes to address. First, the keys can be extracted out of firmware by a sophisticated hacker who has access to the physical device. This has happened in the industry with Link Plugs and other smaller IoT devices [5]. Secondly, these keys can be cloned through replay attacks and deauthentication attacks [4]. This was shown to be effective on all WEP Wifi routers which extracted wifi keys due to the lack of 'freshness' in the messages sent between endpoints and wifi routers. By implementing hardware security primitives such as truly random number generators (TRNGs) and PUFs, many of these prior security vulnerabilities can be drastically mitigated by providing random nonces and authentication signatures. Firstly, the PUFs allow secrets to be embedded in process variations which are not stored in any nonvolatile memory which prevents hackers from simply dumping onboard firmware and finding pre-shared keys. Secondly, the onboard TRNG mitigates replay attacks by preventing attackers from arbitrarily replaying authentication messages.

2.2 MLC NAND flash memory architecture

Flash NAND memory is a type of non-volatile memory that stores user data in the physical form of charge on a *float gate*. Flash NAND memory has exploded in popularity for remote devices and IoT systems because of its high memory density, low cost, and ability to be programmed and erased electronically without moving parts. [6].

Programming these cells requires electrons to move from the polysilicon channel onto or off of the floating gate via electron tunneling, described in Fig. 1A. It is important to note that this electron tunneling can damage the tunnel oxide, which means that flash cells become less reliable after many program/erase cycles (PECs). Most flash memory is rated anywhere from 1000 to 10,000 PECs. A flash's lifetime can increase drastically by using a memory management controller that distributes PECs evenly across all cells in a process called wear leveling. In the case of *Multi-Level Cell (MLC)* flash chips, 2 bits of data are stored on each cell. The float gate voltage is compared to multiple threshold voltages in order to determine the cell's digital value, as shown in Fig. 1B.

2.3 Process variations and PUFs for flash memory

Like all other silicon-based ICs, flash memory chips are subject to many different forms of process variations. In general, most process variations are uncontrollable imperfections caused by limitations in modern lithography processes. These

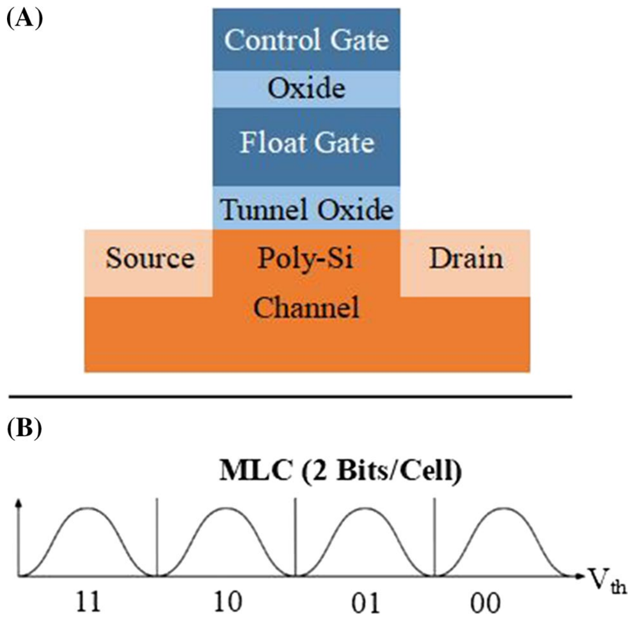


Fig. 1 **A** Overview of NAND flash cell **B** MLC flash chip digitization

variations are commonly caused by various disturbs caused by parasitic capacitance. By performing multiple read or program operations on sections of the flash chips, disturbs can be induced on these chips causing random fluctuations and bit flips [6]. These fluctuations depend on each cell's relative gate thickness and width which are uncontrollable and extremely hard to model. This allows for unique signatures to be generated for each page of the flash memory based on these imperfections. These signatures are known as PUFs and can be used for secure key generation and authentication mechanisms [7]. Each PUF has a challenge and response. The challenge is the input to the function which then outputs an unclonable response. Key mathematical metrics are used to describe PUF efficacy which will be touched on in more detail in Sect. 6.

2.4 Physical unclonable functions (PUFs)

A PUF is a function that produces a unique signature based on challenge-response pairs. These unique signatures often referred to as *silicon fingerprints*, are by-products of intrinsic, uncontrollable process variations found on a silicon-based IC [8]. As mentioned in a previous section, these process variations are present on every chip, and they can greatly differ from one another in regards to performance. While understanding specific instances of process variations can help influence the design for PUF extraction, the absolute variations do not typically matter [9].

3 Related works

Flash memory has gained popularity in recent years due to its cheap cost and high density for storage applications. This is particularly the case for Flash NAND memory. Rather than including extra CMOS as a PUF, utilizing the onboard Flash memory can conserve space and energy for an extremely resource-constrained devices such as those seen in telehealth applications.

When looking at the development of different Flash PUFs, there are several important trends to recognize. The first Flash PUFs were created from 2012–2017 [10–13]. These seminal works were predominately focused on showing how Flash memory was a viable candidate for memory-based PUFs, and they did not consider many of the important design constraints important to Flash PUF technology. From 2018 to 2020, Flash PUF development enhanced and more important practical considerations were considered as seen in the work from [14–20]. Many of these proposed designs significantly enhanced Flash PUF architectures. These Flash PUFs incorporate advanced parameters such as aging resistance, temperature resistance, and unique architectures. For example, work from Clark et al. [15] designed a Flash PUF that was voltage and aging resistant with a novel error correction function. Secondly, Poudel et al. [16] designed a Flash PUF that works on the onboard microcontroller NOR Flash memory. This Flash PUF is one of the first Flash PUFs that requires no periphery hardware and can be used with a variety of NOR flash memory architectures. However, the interrupt process requires many NOP instructions which extends the latency of the PUF. Similarly work from Nguyen et al. [20] also requires minimal periphery hardware and functions on ONFI compliant flash chips. However, it does not discuss aging effects on PUF performance resulting in accuracy degradation as the flash cells reach End of Life. Similarly, Larimian et al. [19] verified the machine learning resilience of their Flash-based PUFs by performing extensive deep learning tests. Furthermore, communication protocols for the flash memory PUF are considered; however, they rely on using traditional cyrptographic tools that may be too costly for resource-constrained microcontrollers. Finally, Mahmoodi et al. [17] created one of the most stable and resilient Flash PUFs by modifying the cells to extract leakage current. Similarly, We et al. [14] create a flash PUF with ultra low error rate throughout the entire lifetime of the flash cells. However, this approach has also requires additional periphery circuitry such as voltage comparators which take up space and require customization of off the shelf flash memory chips. However, the flash cells are modified in order to extract this leakage current. This makes the system not compatible with off the shelf chips.

Several open challenges remain for designing highly stable flash PUF technology that are compatible with off the shelf chips and can be constructed in an economically viable way. In order to make PUFs commercially viable, they must be generated using cheap commercial microcontrollers in order to keep their price point down. Furthermore, it is helpful that these PUFs can be deployed on legacy systems by using commercial off-the-shelf flash memory, and these systems should avoid excessively long latency such as those seen in [10]. Although, some of the most cutting-edge work do not use commercial off-the-shelf components. This can reduce the adoption of Flash PUF technology. Secondly, many of these Flash PUFs require slowly building charge differences on the floating gate through hundred or thousands of program/erase cycles. This is effective at generating signatures; however, this drastically increases the latency required to generate these PUF responses. Finally, many of the systems that avoid the slow build-up of charge have to use expensive FPGAs with Gigahertz clocking speeds to generate fine-grained interrupts as shown in [15]. This is not viable for edge deployments particularly those such as telehealth-based applications.

Our work seeks to bridge this gap by developing a Flash-based PUF that uses only ONFI standard commands; works on a 15 dollar microcontroller; uses off the shelf commodity flash chips; generates signatures with low latency using a novel interrupt technique; and is able to compete with some of the most resilient PUF structures identified in previous work providing a low cost design, strong PUF characteristics, and off the shelf compatibility.

4 HaLo flash PUF extraction technique

HaLo extracts process variations found in flash NAND memory cells for authentication applications. This section will discuss the HaLo experimental setup, design considerations for PUF extraction, and the HaLo method itself.

4.1 Experimental setup

As mentioned in our summarization of contributions section, it is vital to keep the experimental setup as minimal as possible in terms of both cost and complexity. While previous works have mostly required high clock frequencies and custom chips, our novel design uses a 100 MHz clock and a simple TSOP Adapter in order to connect to the flash chip. Our simple bill of materials consists of:

- STM32 based microcontroller with 100 MHz maximum clocking frequency. This will serve as the memory controller and is used to read/write/erase the flash memory.
- 32 GB MLC Flash NAND memory chip from Micron that does not have an integrated memory controller. The memory chip can read/store application data and is also used in the HaLo extraction method
- TSOP Adapter in order to interact with packaged flash NAND chip

The entire experimental setup is purchased for just shy of $20, and all items were readily available for purchase to be included in a commercial application. It is worth noting that we used an unmanaged flash chip in order to give us full control of the device. This extra control allowed us to write directly to specific memory locations without worrying about wear-leveling control or error-correcting code (ECC), which are implemented in most managed flash components. This gave us the highest level of control on interrupting the writing, erasing, and reading processes, because we were directly controlling the chip's behavior without interference from the memory manager. The HaLo method only requires the most basic memory access and modification functions including read page, write page, and erase block.

The experimental setup is shown in Fig. 2. The memory chip interface was *Open NAND Flash Interface (ONFI)* version 2.2, so the interrupt sequence and reading of data are ubiquitous across all ONFI 2.2 memory chips. ONFI 2.2 requires 7 control signals and an 8-bit bus for data, as shown with jumper cables connecting the Flash NAND Chip to the Memory Controller. Additionally, a USB connection was made between the Memory Controller and a computer. PUF trial data was transferred via the USB cable so that all statistical testing could be performed on the computer. A computer is simply an analysis tool, and no part of the actual PUF extraction method is performed on the computer.

The flash chip tested in this work stores pages consisting of 4096 bytes of data and 224 extra bytes for ECC. Blocks are organized in chunks of 256 pages, with each 32 Gb chip having access to 4096 unique blocks.

This setup was built with telehealth applications in mind. Generally speaking, low frequency clocks allow for lower power consumption and lower cost systems. We also worked diretly with an off-the-shelf chip in order to show feasability for industry applications. Unline some previous works mentioned in previous sections, our work can be repeated without the need of custom memory chips or high cost FPGAs.

4.2 Proposed PUF extraction: design considerations

With the experimental setup described in the previous section, work began on the PUF extraction method itself. When designing the PUF extraction method, there are two major metrics that are primarily considered. The first feature was

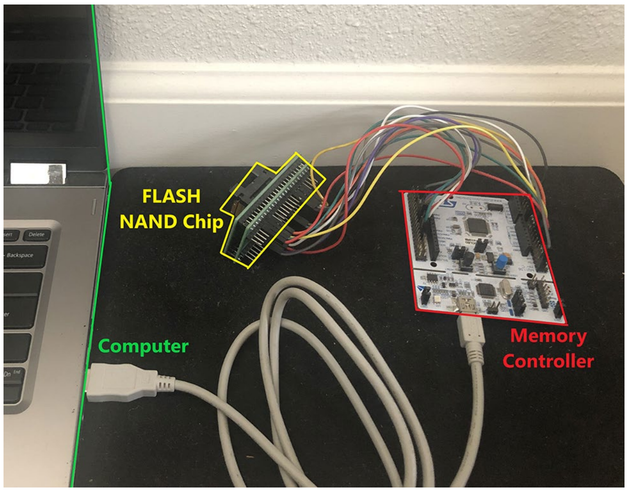


Fig. 2 Experimental setup diagram

minimizing the number of program/erase cycles (PECs) required in order to extract a reliable signature. As mentioned in the *Preliminaries* section, flash memory devices have a limited lifetime measured in PECs. As the charge is tunneled onto and off of the floating gate, the tunnel oxide that separates the silicon channel and the floating gate begins to deteriorate. As the deterioration continues, the data held within each cell becomes unreliable due to the increase of charge leakage on the floating gate. Wear on the cells is an issue for data retention and general application use, but it also poses an issue with unreliable signature extraction, because the programming and erase times for each cell are altered as the cell reaches its end of life. In order to combat this aging effect, HaLo was designed in order to require as few PECs as possible.

The second major consideration was lowering the signature extraction latency. Faster signature extractions allow for faster authentication and lower power consumption. Telehealth sensors in the scope of this work can be treated as edge devices that are resource-constrained by both processor speed as well as limited battery life. In order to prolong the sensor's battery lifetime as well as create a reasonably fast authenticated connection, the method needs to be light-weight and fast.

While additional results and metrics will be discussed in detail in sect. 5 below, these two design considerations helped guide the construction of the HaLo method. In the next section, the HaLo technique will be explained in detail, and additional design constraints and considerations will be discussed.

4.3 PUF extraction observations and techniques

As mentioned in the *Related Works* section, the PUF extraction must have low latency and can only use standard edge deployable microcontrollers within the 100 MHz frequency range. This introduces several important design challenges in order to make Flash PUFs achievable on low-cost microcontrollers. First, the low latency requirement prevents the design from utilizing hundreds of repeated program and read cycles that slowly increase the charge on the floating gloat. Therefore, our scheme must use fine-grained interrupts in order to interrupt the operation of the cells to force the flash into unsteady positions. However, a single interrupt scheme such as the one seen in Clark et al. [15] does not have a high enough clock resolution to interrupt the flash programming fast enough to generate a 50/50 split between ones and zeros. In fact, signatures that are generated from a single programmed interrupt are either about 80% 1's or 80% 0's. This is highlighted in Fig. 3 where the sysTick clock on the microcontroller is interrupted at different times. As shown in the figure, the clock does not have the granularity to interrupt the programming operation on the microcontroller to generate an even distribution of 1's and 0's. This creates two distinct regions detonated by a low programming interrupt and a high interrupt programming interrupt where the low interrupt generates signatures with about 80% ones and the high programming interrupt generates signals with about 80% zeros. The steep drop-off occurs within a single sysTick clock instruction. This makes interrupting the program a distinct challenge.

Secondly, the lack of granularity has another limiting effect as well. The signatures generated from a single interrupt are extremely noisy. This is due to the interrupting method. This error rate can approach 10% within 80 total P/E cycles. This is due to the lack of granularity in the clock itself. Due to the max 100 MHz clocking speed, an interrupt at a particular clocking delay can vary due to the limited frequency. This causes unintended errors for cells that are sensitive

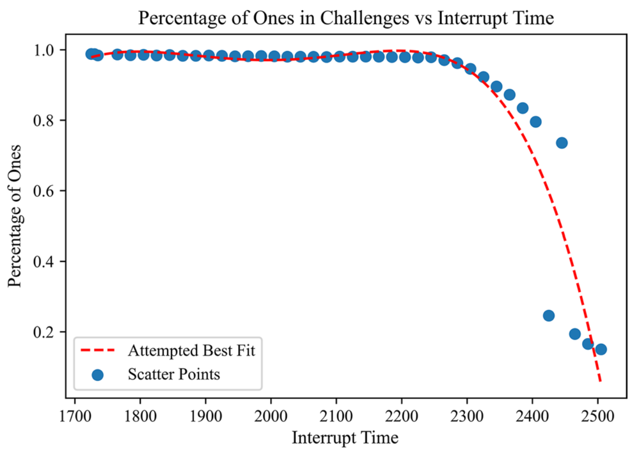


Fig. 3 Ratio of 1's and 0's for different program interrupt times

to the interrupt. This necessitates several programs in the enrollment to ensure that the bytes are stable across a small variation in the actual interrupt delay.

Although the error increases rapidly and the clocking speed cannot generate signatures with 50% ones and zeros, several more insights and solutions were generated to remedy these design challenges. Specifically, a well-defined enrollment scheme is designed to select only the most stable cells which can be decoded as one or zero. These can then be used for highly stably signatures.

4.4 Proposed PUF extraction: enrollment scheme

As other work has shown in Poudel et al. [16] unstable cells can be identified for TRNG bits by applying several reads in rapid succession. These reads apply a smaller voltage to the floating gate of the flash cells which only slightly disturbs the cells. This can quickly identify unstable cells that are flagged during enrollment. Furthermore, approximately 95% of these unstable bits flip within five reads. Therefore, only applying five reads is sufficient for identifying stable cells through successive read operations. Figure 4 highlights this observation. This figure graphically shows all 32000 bits on a single page. All of the yellow lines indicate a flipped bit. Many of these bits flip within the first five reads which is an effective filtering process for identifying stable bits after a single program. Combining multiple reads after each program, and repeating the program and read process multiple times can help identify unstable bits due to lack of clocking granularity.

Secondly, flash cell failure is highly spatially dependent. This is evidenced by plotting a histogram of the distance between each cell flip or failure (Fig. 5). Failures tend to cluster in groups. This can be modeled as a negative exponential distribution between errors. Therefore, instead of flagging individual bits that are stable, bytes are flagged as stable and only if a byte is completely stable is it passed through the enrollment process.

Combining these observations, the first enrollment strategy was crafted. An interrupt program is used on the low side and another is applied to the high side. The low side refers to a program interrupt that creates signatures that are approximately 80% ones. The high side refers to a programming interrupt that creates signatures that are approximately 80% zeros. To be clear, the low side programming interrupt is less than the high side programming interrupt.

A total of five programs with five reads each are performed on the selected page with both the high program and the low program. Since errors tend to be close together, only bytes that are entirely stable are chosen. This creates two separate sets each of which contains 25 signatures due to the five reads per five programs. Next, the stable high bytes locations and the stable low byte locations for a single page are identified. Then bytes that are either highly resistant to programming or highly susceptible to over-programming are chosen. This information is captured by selecting bytes that fully resist programming in the under-programmed section also known as the low side and which bytes are easily programmed in the over-programmed section also known as the high side. If a byte group is simultaneously fully programmed in the over-programmed section and is fully under-programmed in the under-programmed section it is also removed during the enrollment process. This is because greater than 97% of errors are from the aforementioned 700 bytes which are stable in both sections. When referencing Fig. 6, the only byte selected from the low side is Byte 2 and the Byte selected for the high side is Byte 8000. This then creates a 'map' for each page of the stable low bytes and stable high bytes.

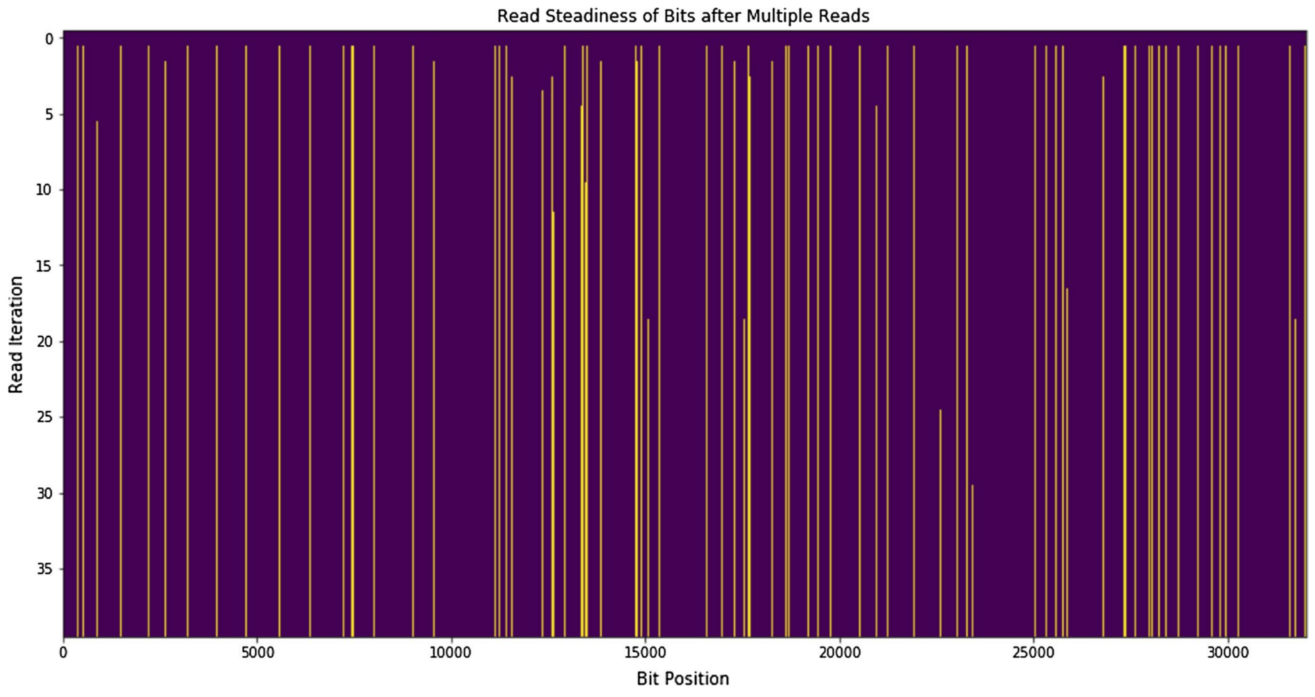
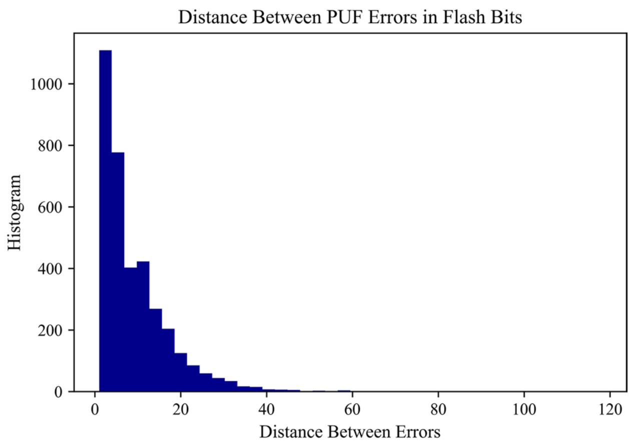


Fig. 4 Error increase over 100 trials

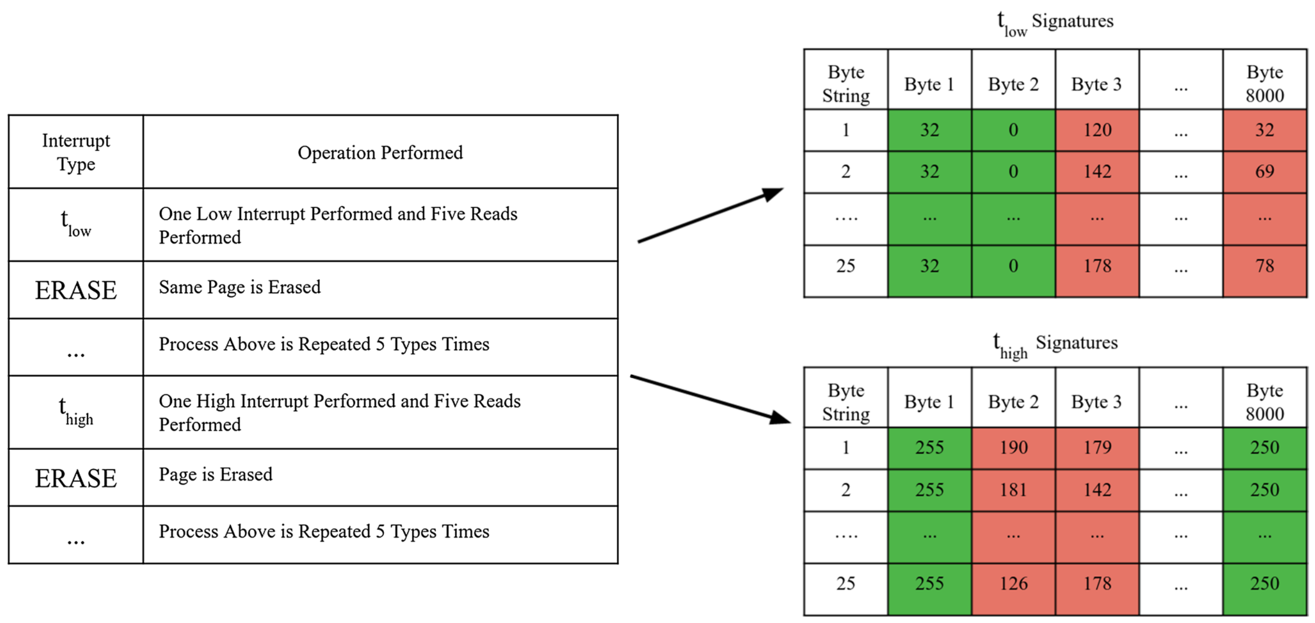
Fig. 5 Histogram of distance between sequential failures



After enrollment, the telehealth device receives a list of byte locations to provide a high program and low program along with the page number to apply the programming. It will do two separate programs: one high and the other low. Then, the majority bit is selected from each byte and is decoded properly. Collisions are minimized since 'mapped' byte values are either highly resistant to over-partial programming or susceptible to it. However, if it occurs the byte with the most amount of parity bits when decoded will be selected. Furthermore, any undefined states (such as equal hamming weights of four) are decoded as low since more errors come from the high side than the low side. This process is highlighted in Fig. 7. This process reduces the percent error by several magnitudes of 10 to around 10^{-4}. A major downside to this extraction process is that it consumes a tremendous amount of bits to extract only the most stable bits and performs two programs instead of one. Specifically, each page consists of 32,000 bits has an output response of about 500 bits.

4.5 Proposed PUF extraction: challenge and response

After the HaLo enrollment process is complete a 'map' of the stable bytes is stored on the gateway. Each byte in the map is either extremely susceptible to over-programming or highly resistant to it. With this information, the



Interrupt Type	Operation Performed
t_{low}	One Low Interrupt Performed and Five Reads Performed
ERASE	Same Page is Erased
...	Process Above is Repeated 5 Types Times
t_{high}	One High Interrupt Performed and Five Reads Performed
ERASE	Page is Erased
...	Process Above is Repeated 5 Types Times

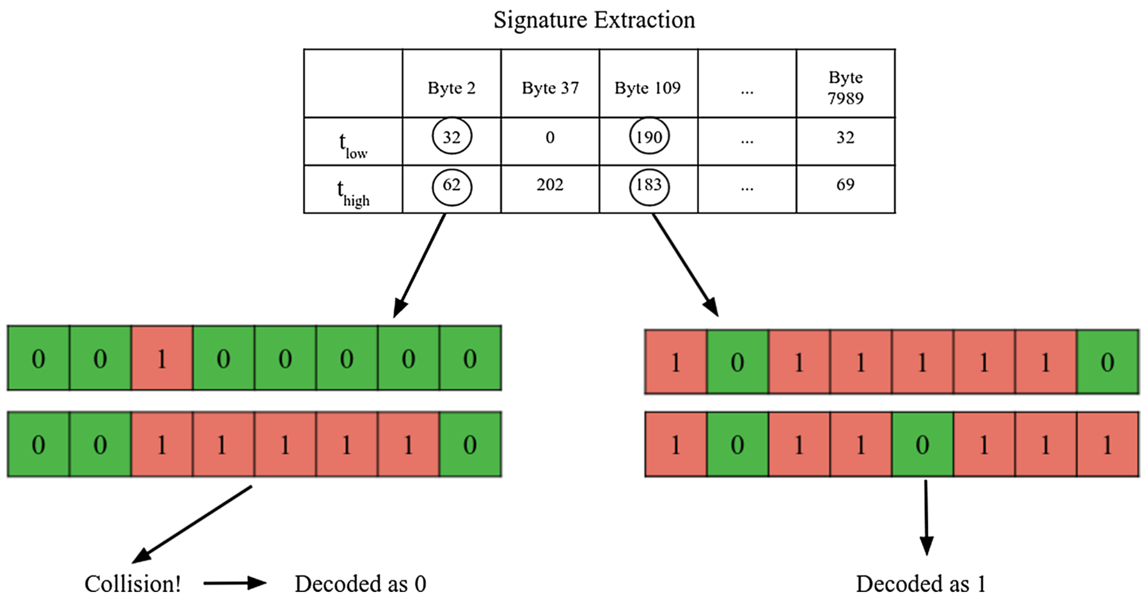
t_{low} Signatures

Byte String	Byte 1	Byte 2	Byte 3	...	Byte 8000
1	32	0	120	...	32
2	32	0	142	...	69
....	...	...	...	...	...
25	32	0	178	...	78

t_{high} Signatures

Byte String	Byte 1	Byte 2	Byte 3	...	Byte 8000
1	255	190	179	...	250
2	255	181	142	...	250
....	...	...	...	...	...
25	255	126	178	...	250

Fig. 6 This figure highlights the enrollment process where low interrupt and high interrupt signatures are generated to provide a 'map' of stable low byte and high byte values. These values are set along with the page number to compromise a challenge



Signature Extraction

	Byte 2	Byte 37	Byte 109	...	Byte 7989
t_{low}	32	0	190	...	32
t_{high}	62	202	183	...	69

Fig. 7 This figure highlights the decoding process for a device generating a response to a PUF challenge. First, the device applies the two programming intervals to the page requested in the challenge. Then, the stable bytes are located and are also sent in the challenge to the end device. The device then applies the programming and chooses the majority represented bit in each signature. In the event of a collision the byte that has the strongest weight is selected. This is why the collision scenario in the figure decodes to 0

gateway will request particular byte locations from the sensor for authentication. Approximately 256 low byte locations and 256 high byte locations will be sent along with the specific page number that is used. This compromises approximately 600 bytes of space which can be sent in the payload of a single Ethernet frame to the sensor. The microcontroller will then apply two programs and ten total reads and identifies which bytes are stable and which are not. If a byte location produces a majority of ones it is subsequently decoded as one and is assumed to be one of the high side bytes. Inversely, if a byte produces a majority of zeros than it is assumed to be from the low side and is decoded as zero. This extraction technique can resist a maximum of three errors before a byte is incorrectly decoded. Therefore, the gateway will receive a bit string of ones and zeros to the sensor.

This design allows for two important features for the gateway. Firstly, the gateway can control how long of an authentication response it needs. This can allow our application to adapt to various levels of required security. For example, certain cryptographic applications may only require 100-bit signatures. The gateway then only has to send 100 bytes to the sensor. On the other hand, sensitive security tasks that may demand 512-bit signatures can also be accomplished. Secondly, many other approaches use helper data such as Hamming Codes, Fire Codes, or more recently Low-Density Parity Codes, to recover any errors sent back from the device. These codes leak polarity information about the response values which allows for error correction. Our scheme leaves out any polarity data and simply sends bytes to read. This makes our helper data significantly more robust to side-channel or modeling attacks since the helper data never reveals any polarity information from the bytes it is requesting. However, it is important to note that this advantage is realized because the HaLo enrollment algorithm filters bytes from pages very aggressively. On average, each page returns approximately 700 usable bytes. Therefore, about 83% bytes are not usable.

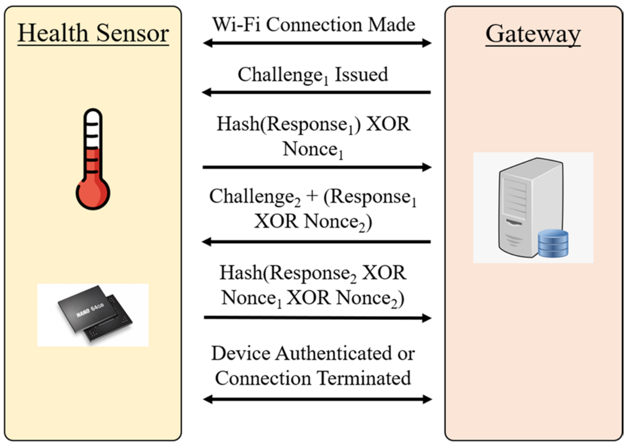
The final step in the HaLo extraction method is authenticating a sensor by issuing a challenge and verifying the response. While most memory based PUF solutions use a challenge-response scheme that links a challenge value with a specific page in memory, HaLo extraction includes additional information in the challenge packet. As mentioned in the *PUF Extraction Observations and Techniques* section, the granularity of the interrupted program is not high enough to generate reliably stable signatures, due to the fact that some cells appear stable in small sample sizes. In order to combat this, the challenge issued to the sensor includes the location in memory, as well as a list of 512 stable byte locations on both the high side and low side of the page programming. It is important to note that these byte indexes do not reveal any polarity of the bytes themselves—they are simply represent a map of bytes that were identified as extremely stable in the enrollment process. The basic authentication procedure is shown in Fig. 8.

5 Experimental results and validations

In this section, we will discuss the relevant PUF metrics of the novel HaLo extraction process. All of the results generated in this section were gathered at room temperature using the experimental setup discussed in the Proposed Flash PUF section. The main characteristics of the HaLo PUF extraction method are its reliability over multiple challenges, the uniqueness of signatures on every page, and the minimal time required in order to generate a signature.

5.1 PUF metrics

There are three major metrics for any PUF. They are uniqueness, randomness, and reliability. Uniqueness defines how different each PUF response is from the other. This value is best captured through the Inter-Hamming Distance (Inter-HD). The percentage Inter-HD describes what percentage of bits flip between two different responses. An ideal value for this is approximately 50%. Our system had an average of 47% which is shown in Fig. 9. The second metric randomness describes how random each signature is. This can be measured in Shannon entropy per bit. With 256 high bits and 256 low bits, the average Shannon entropy per bit is approximately 0.999 with an ideal value of 1. Next, reliability is the final measurement. The reliability of this system is extremely strong. The error does change


Health Sensor
Wi-Fi Connection Made
Challenge$_1$ Issued
Hash(Response$_1$) XOR Nonce$_1$
Challenge$_2$ + (Response$_1$ XOR Nonce$_2$)
Hash(Response$_2$ XOR Nonce$_1$ XOR Nonce$_2$)
Device Authenticated or Connection Terminated
Gateway
Fig. 8 Authentication protocol

as the flash cells age; however, the average percent error is approximately $7.1 * 10^{-6}$. The max percent error value is $1.9 * 10^{-2}$ and the minimum value is $5.9 * 10^{-7}$. This reliability is strong enough to possibly support cryptographic key generation and is able to last through the end of the Flash's life this is shown in Figs. 10 and 11.

5.2 Aging adaptation

In order to simulate aging, the Flash memory chips were programmed until their maximum rating which ranges between 3000 and 4000 P/E cycles for MLC chips. The aging causes the oxide to deteriorate from the program voltage stress. The low side bytes that are susceptible to over programming are barely modified since these bytes just program faster. However, the

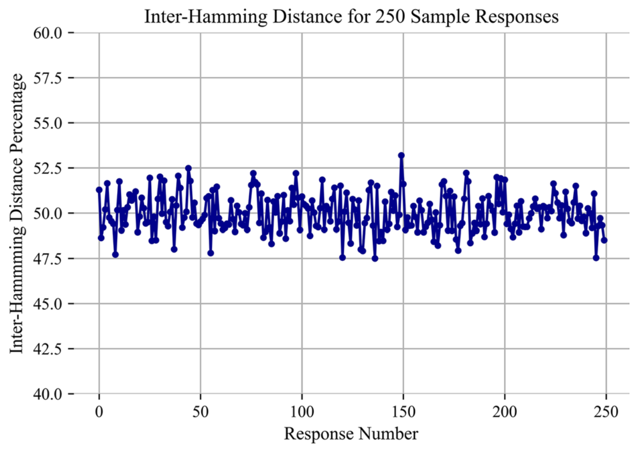


Fig. 9 Inter-Hamming distance calculations between 250 responses

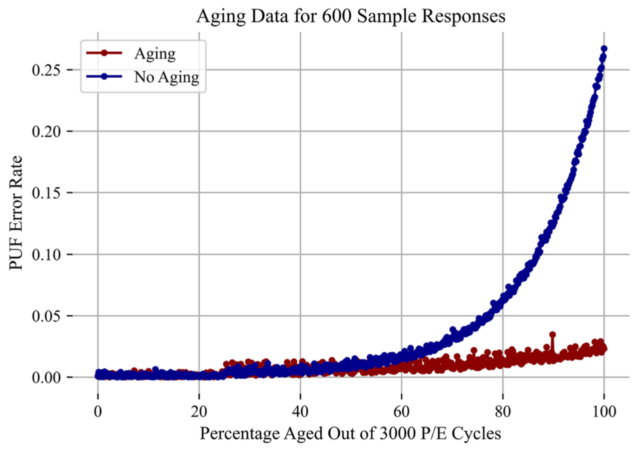


Fig. 10 Error rate vs percentage of life used

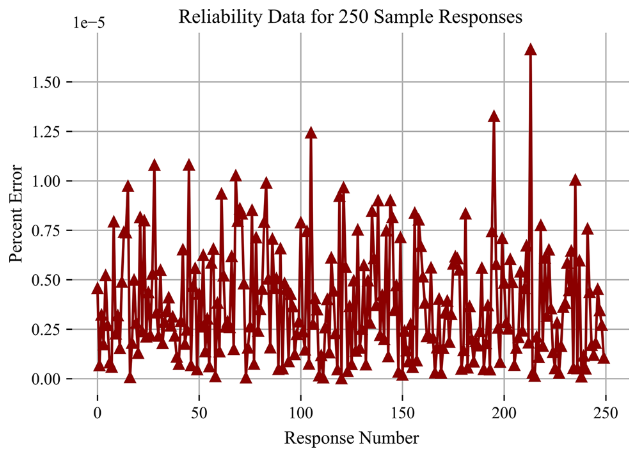


Fig. 11 Reliability of 250 responses at 50% of life used

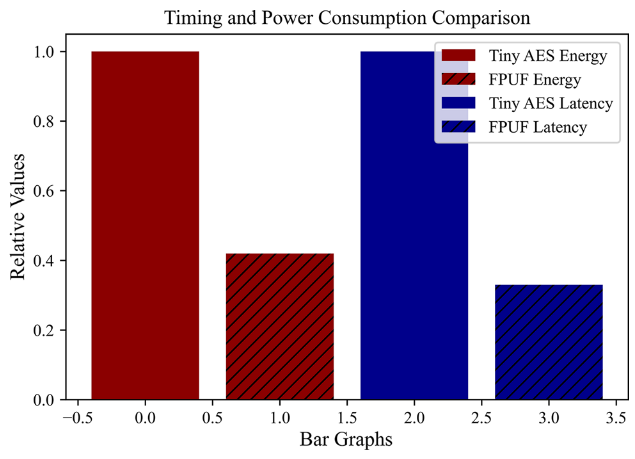


Fig. 12 Relative power consumption and latency comparison between the FPUF and traditional tiny AES encryption scheme

Table 1 PUF entropy, reliability, and randomness metrics

Metric	Minimum	Average	Max
Shannon Entropy	0.99	0.99	0.99
Reliability	$5.9 * 10^{-7}$	$7.1 * 10^{-6}$	$1.9 * 10^{-2}$
Inter-Hamming Distance	47.2%	51.0%	53.1%

high side bytes that are resistant to programming more easily modify the amount of bit flips in each byte. In general two approaches for this were considered. First was adapting the polling interval and decrease it since the flash cells program faster. However, this approach can be difficult to model and control due to the lack of granularity in the interrupt mechanism. Consequently, the second approach was taken. In this approach, the number of bytes required for a decode was changed once the cells reached a particular percentage of life used. This adapts the error rate and drops it significantly. Instead of taking the majority (or in our case 5 bits) for a decode. The decode threshold was varied according to the life of the flash cells. At 50% of the lifetime, the amount of bits required for a proper decode becomes 6. Then, this changes to a threshold of 7 at 90% which remains fixed and keeps the error rate significantly lower. This trend is reflected in Fig. 10. All of the aggregate statistics are reflected in Table 1.

5.3 PUF latency and power consumption

Finally, our PUF will be compared in latency and power consumption to an AES encryption scheme on one of the most portable AES implementations for resource-constrained devices, Tiny AES. To just encrypt a preshared key about 22.3 mW of power is used and the encryption process takes around 190 ms. However, the HaLo extraction technique takes less than 1mW of power and PUF signatures are generated in 34.8 ms. This relative power difference is shown in Fig. 12. This highlights a significant performance advantage with a higher security guarantee since the PUF signatures are never stored anywhere. Furthermore, this low latency and power consumption make the HaLo PUF a strong candidate for the telehealth application space.

6 Telehealth application

The first step in developing our authentication protocol was to build our telehealth application, which involved using a DS18B20 temperature sensor to collect temperature data and store it in a format that we can then use for edge deployment. Body temperature collection is just one of the many different ways remote patient monitoring can be utilized. For our experiment, we used a Raspberry Pi as a means to collect temperature data in intervals of 10 s and saved the data in a csv format.

We then built a TCP/IP dynamic challenge-response authentication scheme using the PUF. This authentication process starts with the gateway sending a challenge to the health sensor, denoted Challenge_1. This request includes a challenge location and stable byte locations of the NAND flash memory cell. The health sensor performs the interrupted program and extracts the stable byte values. The health sensor then randomly generates a nonce, denoted Nonce_1, and the generated hash value is XORed with Nonce_1. The value generated is sent to the gateway as a response to Challenge_1. The gateway knows what Response_1 should be, so it XORs the entire response of the gateway with Response_1 to determine Nonce_1. The gateway then randomly generates another nonce, denoted Nonce_2. A second challenge, using a different index of stable byte locations, is generated by the gateway. Challenge_2 is then concatenated with (Response_1 XOR Nonce_2), and this message is sent back to the health sensor. The health sensor is able to separate Challenge_2 from (Response_1 XOR Nonce_2), and (Response_1 XOR Nonce_2) is XORed with Response_1 to determine Nonce_2. Both the gateway and the health sensor now know the values of Challenge_1, Challenge_2, Nonce_1, and Nonce_2. Using Challenge_2, the health sensor again performs the interrupted program and extracts the stable byte values. By the end, if the health sensor and the gateway are legitimate sources, the challenges and responses can be properly decoded and the transaction is authenticated.

There are many advantages to using this authentication scheme. First, the use of hashes masks the plaintext values of the data being sent, meaning that attackers won't be able to read the data in transit. Second, the use of randomly generated nonces means that the values being transmitted will always change with every transaction. Finally, and most importantly, the use of a PUF provides a unique identifier that the gateway can reliably authenticate, and attackers won't be able to model the authentication responses of the PUF.

Here we see the challenge and response pairs that we discussed being sent between the health sensor and the gateway. In the event of a man-in-the-middle or replay attack, which involves an attacker intercepting packets between the two parties, this is the information that they would receive (apart from some of the comments I wrote here). However, with regard to MITM, any modification of the packets being sent would result in a complete breakdown of the authentication process, ultimately making it obvious that the connection was tampered with, and would result in the gateway refusing the connection. With Replay attacks, the use of randomly generated nonces means the values being sent change with every transaction. This means that an attacker can't replay a message with an old pair of nonces, as the values would be entirely different to the current nonces.

7 Conclusions and future works

As medical care continues to improve in both effectiveness and accessibility on a global scale, remote patient monitoring is a logical next step for healthcare investment. We have witnessed a global pandemic that disrupted the delivery of potentially life-saving medical care for over a year, which may have been partially mitigated by the adoption of remote patient monitoring (RPM) technology. While the technology proves useful for many medical monitoring applications, it is important to carefully consider the security vulnerabilities and possible exploits of large-scale adoption of RPM. The HaLo extraction method is a lightweight, fast, and easily implementable security measure that could help ensure the authenticity of RPM sensor data. The HaLo method is designed in such a way that it could be implemented on new RPM sensors with minor software updates, and no additional components. The method is resource-efficient and offers financial benefits when compared with many other commercial PUF solutions.

When considering future work for this project, there are many avenues which can be explored. One obvious avenue is exploring more flash based technologies such as single level cell (SLC), triple later cell (TLC), quad level cell (QLC), and 3-D based architectures. Proving the HaLo method's viability on multiple architectures of flash chips can further prove the usefulness of the method by providing additional memory chip options for IoT system designers. These architectures could also possibly offer improved metrics in latency, power consumption, or reliability.

Another area of research could stem from the unsteady cells in each page. While PUF signatures require high degrees of reliability in order to produce robust keys used for authentication, there is a possibility of producing a truly random number generator (TRNG) by reading unsteady bit positions. No statistical testing was done on this topic so it is only a hypothesis at this point, but a method that produces a TRNG and PUF signature could be heavily sought after in the IoT field.

Finally, there should be additional testing done in order to ensure robustness of the HaLo method with variable environmental conditions. One major conditions important to IoT devices would be temperature tolerance. Depending on the application, some IoT devices could be subjected to unusually high or low temperatures, but the reliability must remain high. Another factor could be flash memory chip manufacturer. Although in principle this method could be directly translated to any MLC flash NAND chip, the internal circuitry and interface of each manufacturer could change results.

Authors' contributions HG did the analysis on the experimented data, and created the plots. TL collected the data from Flash memories, and validate them. CK did the authentication part and helped with the paper writing. ST (Advisor) evaluated the results, and helped with writing the paper. All authors read and approved the final manuscript.

Data availibility Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Declarations

Competing interests The authors declare no competing interests.

References

1. Camara C, Peris-Lopez P, Tapiador JE. Security and privacy issues in implantable medical devices: a comprehensive survey. J Biomed Inf. 2015;55:272. https://doi.org/10.1016/j.jbi.2015.04.007.
2. Palve A, Patel H, Towards Securing Real time data in IoMT Environment, In: 2018 8th International Conference on Communication Systems and Network Technologies (CSNT) 2018;113–119. https://doi.org/10.1109/CSNT.2018.8820213
3. Fakroon M, Gebali F, Mamun M. Multifactor authentication scheme using physically unclonable functions. Internet Things. 2021;13:100343. https://doi.org/10.1016/j.iot.2020.100343.
4. Berghel H, Uecker J. WiFi attack vectors. Commun ACM. 2005;48:21. https://doi.org/10.1145/1076211.1076229.
5. Ling Z, Luo J, Xu Y, Gao C, Wu K, Fu X. Security vulnerabilities of internet of things: a case study of the smart plug system. IEEE Internet Things J. 2017;4(6):1899. https://doi.org/10.1109/JIOT.2017.2707465.
6. Gordon H, Edmonds J, Ghandali S, Yan W, Karimian N, Tehranipoor F. Flash-based security primitives: evolution, challenges and future directions. Cryptography. 2021;5:1. https://doi.org/10.3390/cryptography5010007.
7. Luo Y, Ghose S, Cai Y, Haratsch EF, Mutlu O. Improving 3D NAND flash memory lifetime by tolerating early retention loss and process variation. Proc ACM Measure Anal Comput Syst. 2018;2(3):1.
8. Xu SQ, Yu Wk, Suh GE, Kan EC, Understanding sources of variations in flash memory for physical unclonable functions, In: 2014 IEEE 6th International Memory Workshop (IMW) 2014;1–4. https://doi.org/10.1109/IMW.2014.6849385
9. D.E. Holcomb, W.P. Burleson, K. Fu, Initial SRAM state as a fingerprint and source of true random numbers for RFID tags, In: Proceedings of the Conference on RFID Security 2007;
10. Prabhu P, Akel A, Grupp LM, Wing-Kei SY, Suh GE, Kan E, Swanson S, Extracting device fingerprints from flash memory by exploiting physical variations, In: International Conference on Trust and Trustworthy Computing (Springer, 2011), 188–201
11. Wang Y, Yu Wk, Wu S, Malysa G, Suh GE, Kan EC. Flash memory for ubiquitous hardware security functions: True random number generation and device fingerprints, In *2012 IEEE Symposium on Security and Privacy* (IEEE, 2012), pp. 33–47
12. Jia S, Xia L, Wang Z, Lin J, Zhang G, Ji Y, Extracting robust keys from nand flash physical unclonable functions, In: International Conference on Information Security (Springer, 2015), pp. 437–454
13. T.S. et al., High-temperature stable physical unclonable functions with error-free readout scheme based on 28nm SG-MONOS Flash Memory for Security Applications, (IEEE, 2017), pp. 1–4

14. Wu M, Yang T, Chen L, Lin C, Hu H, Su F, Wang C, Huang JP, Chen H, Lu CC, Yang EC, Shen RS. A PUF scheme using competing oxide rupture with bit error rate approaching zero, In: 2018 IEEE International Solid - State Circuits Conference - (ISSCC) (2018), pp. 130–132. https://doi.org/10.1109/ISSCC.2018.8310218
15. Clark LT, Adams J, Holbert KE. Reliable techniques for integrated circuit identification and true random number generation using 1.5-transistor flash memory. Integration. 2019;65:263.
16. Poudel BRP, Milenkovic A. Microcontroller TRNGs using perturbed states of NOR flash memory cells. IEEE Trans Comput. 2019;68:307–13.
17. Mahmoodi M, Nili H, Larimian S, Guo X, Strukov D, ChipSecure: A reconfigurable analog eFlash-based PUF with machine learning attack resiliency in 55nm CMOS, In: 2019 56th ACM/IEEE Design Automation Conference (DAC) 2019;1–6
18. Sakib S, Milenković A, Rahman MT, Ray B. An aging-resistant NAND flash memory physical unclonable function. IEEE Trans Electron Devices. 2020;67(3):937.
19. M.R.M. S. Larimian, D.B. Strukov, Lightweight integrated design of PUF and TRNG security primitives based on eFlash memory in 55-nm CMOS, (IEEE, 2020), 67:1586–1592
20. Nguyen TN, Park S, Shin D. Extraction of device fingerprints using built-in erase-suspend operation of flash memory devices. IEEE Access. 2020;8:98637. https://doi.org/10.1109/ACCESS.2020.2995891.