Balancing Security and Usability of Zero-interaction Pairing and Authentication for the Internet-of-Things

Kyuin Lee
University of Wisconsin–Madison
kyuin.lee@wisc.edu

Younghyun Kim
University of Wisconsin–Madison
younghyun.kim@wisc.edu

Abstract

The dramatic increase in the number of connected and Internet-of-Things devices today has called for more usable authentication methods to establish a secure connection among a large number of spatially dispersed devices with minimal or no human involvement. Recently, zero-interaction pairing and authentication (ZIPA) has emerged as a promising solution to this challenge, and researchers have devised various techniques to exploit different entropy sources to generate keys and set up secure connections. While prior works have successfully shown the potential of ZIPA, the focus has been placed mainly on exploring new sensing modalities, but barely on optimizing the underlying signal processing that is crucial for balancing security and usability. In this paper, as a first step towards the systematic design optimization of the signal processing pipeline of ZIPA techniques, we propose a generic key reconciliation framework that determines the proper reconciliation parameter based on a user-given authentication range. We analyze and compare the two most commonly used reconciliation schemes in terms of security and usability to select the better scheme with the best parameters to set the balance between them and estimate the computation overhead.

CCS Concepts

• **Human-centered computing** → **Ubiquitous and mobile computing systems and tools**; • **Security and privacy** → *Authentication.*

Keywords

Zero-interaction pairing and authentication; Context-based authentication; Key reconciliation; Usable authentication method

ACM Reference Format:
Kyuin Lee and Younghyun Kim. 2021. Balancing Security and Usability of Zero-interaction Pairing and Authentication for the Internet-of-Things. In *Proceedings of the 2nd Workshop on CPS&IoT Security and Privacy (CPSIoTSec '21), November 15, 2021, Virtual Event, Republic of Korea.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3462633.3483977

CPSIoTSec '21, November 15, 2021, Virtual Event, Republic of Korea

ACM ISBN 978-1-4503-8487-2/21/11…$15.00
https://doi.org/10.1145/3462633.3483977

1 Introduction

The number of connected and Internet-of-Things (IoT) devices has been experiencing explosive growth, driven by emerging applications and advanced technologies, coupled with the active standardization of connected ecosystems. As the number is continuing to increase, efficient configuration and long-term management of these devices are becoming more time-consuming and labor-intensive. For instance, current *pairing* (mutually registering two devices with no prior knowledge) and *authentication* (verifying the authority of a device to access resources) methods between typical IoT devices (e.g., smart speakers, lights, and thermostats) heavily involves *human-in-the-loop* operations by requiring the user to manually type in a pin or password to establish credentials between two devices [2]. When the devices do not have a proper user interface, which is typical for small, low-cost devices, the procedure is further complicated through the usage of a secondary device, mainly the user's smartphone. This cumbersome procedure often puts a burden on the user not only during the pairing time but also when re-authenticating existing devices (e.g., periodic password update), which results in inexperienced users opting not to change default or old passwords. The lack of a usable pairing and authentication scheme has become an imminent threat to IoT systems as disclosed in the 2016 Data Breach Investigations Report — 63% of confirmed data breaches are attributed to using weak, default, or stolen passwords [13].

The efforts to reduce human involvement have lead researchers to introduce various *zero-interaction pairing or authentication (ZIPA)* techniques. ZIPA techniques exploit a *spatially correlated, temporally uncorrelated environmental noise* to allow devices to authenticate with each other only when they are closely located at the same time by independently deriving identical keys from the noise. The main assumption that underlies ZIPA is that if two devices independently observe substantially correlated environmental noise patterns, they are likely to be located in the same physical space called an *authentication range*, which implicitly means that they are owned and controlled by the same user, and hence can be allowed to authenticate with each other. Since the noise is significantly different when measured at different locations, distant devices outside the authentication range cannot generate the same key. Compared to traditional password-based authentication schemes, ZIPA is advantageous in terms of both security and usability. The random keys provide higher security than user-created passwords do, and it does not depend on the user to create, remember, and enter the password. Furthermore, it allows devices to use a unique key for each device pair or re-establish fresh keys more frequently and autonomously.

While various ZIPA techniques have been proposed in many different works, the focus has mainly been placed on the modalities

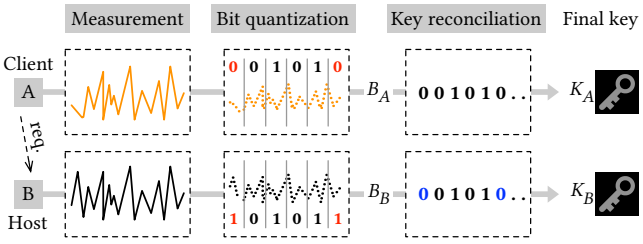

Figure 1: General pipeline of ZIPA techniques between two Devices: A (Client) and B (Host).

of sensing, i.e., the sources of entropy to create the keys. However, the signal processing pipeline that generates the keys from the measurement has received less attention and has been rather ad hoc. The pipeline typically consists of stages for measurement, bit quantization, and key reconciliation, which are highly tailored and designed toward a specific sensing modality. A signal processing stage designed for one technique does not work for others, and the lack of a "systematic" pipeline design is a large missing piece of ZIPA techniques.

In this paper, as a first step towards exploring a systematic approach for signal processing in ZIPA techniques, we investigate and focus on the key reconciliation stage among other stages that is most sensitive to parameter tuning and most computationally heavy. More specifically, we propose a generic framework that automatically determines the reconciliation parameter that balances security and usability, given a user-defined authentication range. The proposed framework can serve as a guideline for ZIPA developers when coming up with new techniques or be implemented in existing ZIPA works to seamlessly determine proper parameter values. Additionally, we analyze the two most commonly used key reconciliation schemes (leveraging error-correcting codes and compressed sensing) in terms of their reconciliation rates, entropy loss, and computation costs based on varying parameters to determine a standard scheme to be used in our framework.

2 Security and Usability of ZIPA

2.1 Signal processing pipeline

The security and usability of a ZIPA technique are sensitive to the design of its signal processing pipeline. A typical pipeline of ZIPA techniques is illustrated in Figure 1. The pipeline consists of three general stages ("Measurement", "Bit quantization", and "Key reconciliation") to generate the final authentication key. We assume that Device A (Client) initiates the process by sending an authentication request to Device B (Host).

Measurement: Two devices independently measure a surrounding noise context using their embedded sensors.

Bit quantization: The noise measurements are quantized into a sequence of 1s and 0s using statistical features in the time or frequency domain. We refer to the quantized bit sequences as *environmental bit sequences*, B_A and B_B, and further denote their length with b. Even the bit sequences from two very closely located devices will likely exhibit occasional bit errors due to various factors including sensor variations, random errors, etc.

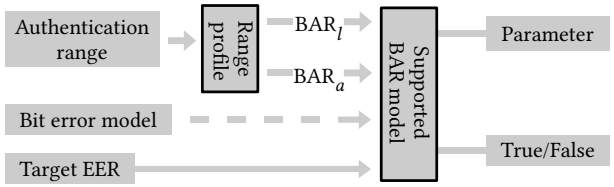

Figure 2: Framework to determine reconciliation parameter given three user inputs: authentication range, bit error model (optional), and target EER.

Key reconciliation: The bit errors between B_A and B_B are reconciled using a reconciliation scheme to produce identical *final keys*, K_A and K_B, of bit length k. During the reconciliation, partial information about B_A and B_B are exchanged over the public channel that incurs some entropy loss which results in K exhibiting less entropy than B. It is important to ensure that the exchanged information does not leak enough information to fully derive B nor K because the public channel is always assumed to be eavesdropped by malicious adversaries trying to infer K.

2.2 Balancing security and usability

Authentication is deemed successful only when the two resulting final keys are identical with no bit errors. We define all devices within the authentication range from each other as legitimate devices and further refer to devices located outside the range as adversarial devices. Note that the only difference between the legitimate and adversarial devices is whether they are located in or outside the authentication range. As a metric to compare the similarity of two bit sequences (i.e., B or K), we use *bit agreement rate* (BAR), which refers to the rate of matching bits. To quantify usability, we use *true acceptance rate* (TAR), referring to the rate of successful authentication between the legitimate devices. The security of the system is quantified with *false acceptance rate* (FAR), which is the rate of successful authentication of adversarial devices. The overall usability and security balance is measured with *equal-error rate* (EER), which refers to the intersection point where FAR is equivalent to false rejection rate (FRR=100−TAR). A low ERR represents better accuracy of distinguishing legitimate devices over adversarial ones.

Because a device's physical location is a proxy for its legitimacy in ZIPA, it is important to be able to strictly control the authentication range at which devices should be allowed to authenticate with one another. The parameters of the key reconciliation scheme determine its error tolerance threshold, which in turn depicts the boundary of the authentication range after two authenticating devices translate its measured environmental noise into bit sequences; if and only if the two bit sequences exhibit fewer errors than the threshold for the given parameters, identical keys can be derived to authenticate two devices. As such, the reconciliation parameters should be carefully chosen because the parameters that correspond to a high error tolerance threshold will allow distant adversarial devices to be able to authenticate (high FAR), while a too low threshold will suffer from low usability (high TRR). To strike this sensible balance between security and usability, it is crucial to understand and determine proper reconciliation parameters for different existing reconciliation schemes.

Table 1: Error-correcting code based reconciliation (Fuzzy commitment) [2, 3, 6, 10–12]

	Device A		Device B
①	$K_A = PRNG(k)$		
②	$H(K_A) = SHA256(K_A)$		
③	$\lambda_A = ENCODE(K_A)$		
④	$\sigma = B_A \oplus \lambda_A$	$\overset{\sigma, H(K_A)}{\longrightarrow}$	$\lambda_B = B_B \oplus \sigma$
⑤			$K_B = DECODE(\lambda_B)$
⑥			$H(K_A) \overset{?}{=} H(K_B)$

Table 2: Compressed sensing based reconciliation [7, 8, 14, 15]

	Device A		Device B
①	$K_A = B_A$		
②	$H(K_A) = SHA256(K_A)$		$C_B = \Phi \cdot K_B$
③	$C_A = \Phi \cdot K_A$	$\overset{C_A, H(K_A)}{\longrightarrow}$	$\Delta C = C_A - C_B$
④			$\Delta B = \text{l1.min}(\Delta C)$
⑤			$K_B = B_B \oplus \Delta B$
⑥			$H(K_A) \overset{?}{=} H(K_B)$

Different ZIPA techniques use different forms of environmental noise with different spatial extents. Generally, as the distance between the two devices increases, BAR between their environmental bit sequences decreases due to spatially unique nature of the noises. For instance, in VoltKey, which utilizes powerline noises as a source of entropy, BAR decreases from 95% to 91% as the distance between the devices increases from 1 m to 24 m [4]. To balance security and usability for a given authentication range, the key reconciliation parameters must be judiciously fine-tuned to successfully authenticate only when BAR is higher than or equal to the BAR of the desired range. While most previous works presented new methods and different noise sources that ZIPA can leverage, the problem of automatically determining optimal reconciliation parameters for varying authentication range is yet to exist.

We propose a systematic framework to analyze and determine the parameters given an *authentication range* and a *target EER* defined by either the manufacturer or the user. The overall diagram of the framework is illustrated in Figure 2. As inputs, it takes in the desired authentication range and a target EER of the system. The bit-error model between B_A and B_B (e.g., burst error model) is an optional input. With the desired range, the "Range profile" block determines the minimum BAR that is required between the legitimate devices, BAR_l, and the maximum BAR that is exhibited by devices outside the range, BAR_a. The Range profile block is pre-profiled with the varying distance and its corresponding BAR between B_A and B_B, which may differ from one noise source to another. Once profiled, BAR_l is selected as the corresponding BAR under the given range, and BAR_a is selected as the highest BAR achieved beyond the range. With the selected BAR_l and BAR_a, the "Supported BAR model" block, which is also pre-profiled with varying reconciliation parameters and its reconciliation performance, outputs the optimal parameter. Additionally, it confirms if the user-given target EER can be met. Note that this block can be based on different existing key reconciliation schemes leveraging error-correcting codes or compressed sensing, which will be described in the next section. This framework can particularly benefit ZIPA developers to quickly determine proper reconciliation parameter (regardless of noise sources) that balances usability and security (EER) without manually computing through different parameter values and validating the EER point.

3 Key Reconciliation Protocols

In this section, we describe two commonly used key reconciliation schemes in ZIPA research to understand how their parameters affect the performance of ZIPA techniques. One scheme is based mainly on *error-correcting code (ECC)* and the other leverages the notion of *compressed sensing (CS)* to reconcile differences in bit sequences.

3.1 ECC-based reconciliation

ECC-based reconciliation, also commonly known as *fuzzy commitment* scheme, leverages linear ECC to allow two very similar bit sequences as evidence to lock and unlock a "secret" so that it can be transferred over public wireless channel [3]. In our case, the "secret" is the final key, K, and the two environmental bit sequences from two devices are used to lock and unlock K.

The overall reconciliation steps are illustrated in Table 1. First, Device A uses a pseudo-random number generator to generate a random bit sequence of length k to be used as a final key, K_A. It then performs one-way hashing on the result (e.g., SHA256) to generate $H(K_A)$. Afterward, K_A is encoded into a codeword, λ_A, of length b using an ECC (e.g., Reed-Solomon(b,k)) to add redundancy based on the polynomials over Galois fields to support error correction. Then, it performs exclusive OR between λ_A and extracted B_A to obtain σ, which is transferred to Device B through a public channel along with $H(K_A)$. Note that σ does not divulge any information about B_A nor λ_A to the malicious eavesdropper on the public channel. Once Device B receives σ, it performs exclusive OR with its own extracted B_B to produce λ_B which is very close to originally encoded λ_A. Afterward, λ_B is decoded using the same ECC into K_B. To verify the equality of the two keys obtained on two devices, K_A and K_B, Device B compares the two hashed results; if equal, two devices are successfully authenticated. The fuzzy commitment scheme assumes identical derivation of the key as long as the number of bit errors (Hamming distance) between B_A and B_B does not exceed $\frac{b-k}{2}$ bits, and we denote this number of bit error as T. Note that the added redundancy to correct the error between the B_A and B_B results in final key length k to be shorter than b ($k < b$). In Section 4, we use T as a main parameter to benchmark the ECC-based reconciliation.

3.2 CS-based reconciliation

CS is an information processing technique to reconstruct a high-dimensional sparse signal from a low number of measurements. A CS-based reconciliation scheme mainly uses this property to

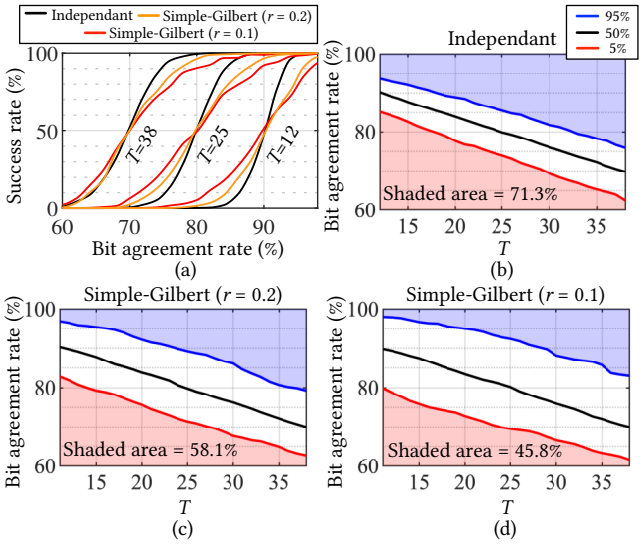

Figure 3: (a) Success rate for ECC-based scheme varying T and required BAR between devices to achieve 5%, 50% and 95% success rate in (b) independent and (c) simple Gilbert model with r=0.2 and (d) r=0.1.

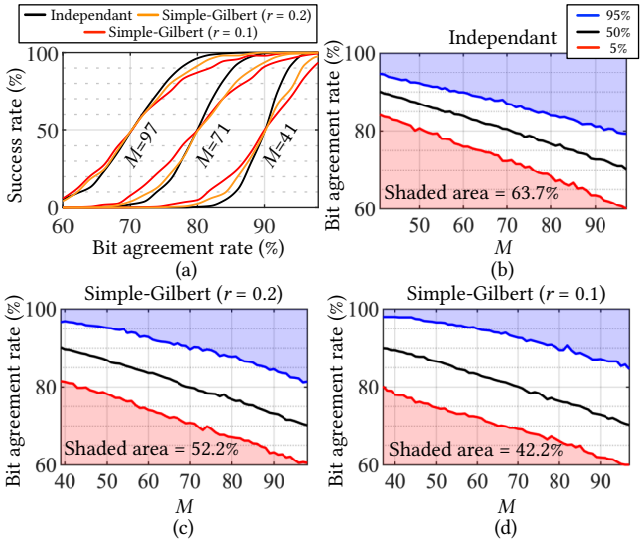

Figure 4: (a) Success rate for CS-based scheme varying M and required BAR between devices to achieve 5%, 50% and 95% success rate in (b) independent and (c) simple Gilbert model with r=0.2 and (d) r=0.1.

reconstruct the sparse error between B_A and B_B as they are very close to each other in terms of their Hamming distance.

The overall steps are illustrated in Table 2. Unlike ECC-based reconciliation, Devices A and B directly use the extracted B_A and B_B to be used as K_A and K_B, respectively, which results in $k = b$. Device A further obtains the hashed key to later verify the equality with K_B. Then, K_A and K_B are compressed into C_A and C_B, respectively, with Φ representing the random Bernoulli matrix with ± 1 with equal probability. The Φ represents a randomly generated sensing matrix that is assumed to be identical on both devices (can be pre-established) and the dimension of Φ specifies the compression rate; $k \times M$ matrix results in the length of C_A to be M bits. Afterward, Device A transfers C_A, along with $H(K_A)$ to Device B. Upon receiving, Device B obtains ΔC by subtracting the two C which essentially represents the difference between B_A and B_B in a lower dimension. Afterward, the difference, ΔB, is recovered to a higher dimension signal by solving the l_1 minimization problem of ΔC, which regularizes the problems by using the sparsity of the solution. Then, K_B can be derived by performing exclusive OR between ΔB and B_B. In Section 4, we use M as a main parameter to benchmark the CS-based reconciliation.

4 Analysis of Key Reconciliation Schemes

In this section, we benchmark the two above-mentioned key reconciliation schemes in terms of their reconciliation performance, entropy losses, and computation overheads. For all evaluations, we assume $b = 128$ bits and vary the parameters T and M for ECC- and CS-based reconciliation, respectively.

4.1 Reconciliation success rate

We first compare how the key reconciliation parameter affects the performance of the two reconciliation schemes. As an evaluation metric, we use *success rate*, which refers to the rate of successful authentication (100% BAR between K_A and K_B) to all authentication attempts. To simulate two environmental bit sequences, B_A and B_B, with varying BAR, we first use a pseudo-random number generator to obtain B_A and consider three error characteristics to obtain B_B with two different bit error models: the *independent* model and the *simple Gilbert* model [1]. In the independent error model, every bit has an equal bit error rate equivalent to the given bit error rate (i.e., 100%-BAR). In the simple Gilbert model, the bit errors are based on the two-state (good and bad states) Markov approach, where each of them may generate errors based on its state (good: 0%, bad: 50%). We use r (the probability of transitioning from bad to good state) of 0.1 (more bursty) and 0.2 (less bursty) to simulate different error burstiness. Using the three different error characteristics, we vary the BAR from 60% to 98% on a sequence of 100,000 bits.

Figures 3(a) and 4(a) illustrate the success rate of ECC- and CS-based reconciliation, respectively, for varying parameters. Since BAR monotonically decreases as the inter-device distance increases, an ideal success rate curve should have a steep cut-off slope at the BAR that corresponds to the target distance. This would allow devices with a BAR above the threshold to always succeed in authenticating and reject devices with a BAR below the threshold. However, in reality, the success rate curve show a gradually sloping curve because the BAR distribution of B within the specified mean BAR exhibit a normal distribution. In all cases, a burstier error results in a flatter slope. This is because the BAR variances of each B with $b = 128$ are higher for burstier error models even though all three error characteristics exhibit an identical mean BAR. In both schemes, as the error tolerance parameter, T or M, increases, the success rates increase for given BAR. This means that an increase in T and M can tolerate more number of bit errors between devices. Specifically, on ECC-based reconciliation, $T = 12$ achieves a 50% success rate when the BAR between the devices is

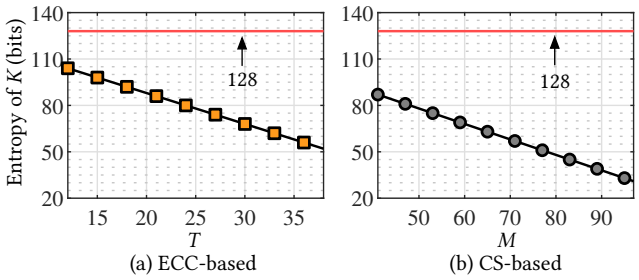

Figure 5: Entropy of the final key, K, using (a) ECC-based and (b) CS-based reconciliation schemes.

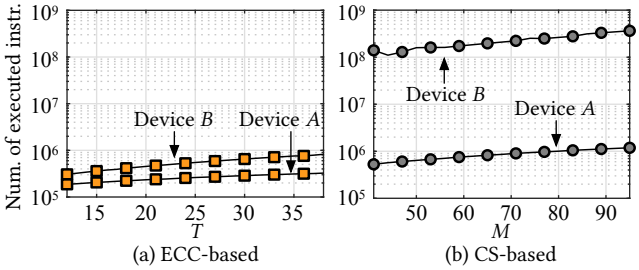

Figure 6: Number of executed instructions under (a) ECC-based and (b) CS-based reconciliation schemes.

at 90% under different error characteristics. As T increases to 38, the same 50% success rate is achieved when the two devices only exhibit a 70% BAR. Similar performance is achieved on CS-based reconciliation with $M = 41$ and 97 that can support 90% and 70% BAR, respectively.

Figures 3(b), (c) and (d) illustrate the required BAR between devices to achieve 5%, 50% and 95% success rate for varying parameters (equivalently supporting 70% to 90% BAR) for each error characteristics with the EEC-based reconciliation; and Figures 4(b), (c) and (d) illustrate the same for the CS-based reconciliation. For each parameter, T or M, the shaded area above 95% line (blue) indicates the supported BAR between legitimate devices (BAR_l) to achieve the minimum TAR of 95%. On the other hand, the shaded area below 5% line (red) indicates the supported BAR from the adversarial devices (BAR_a) to achieve a maximum FAR of 5%. This indicates that as long as BAR_l is above the blue line and BAR_a is below the red line for a specific parameter, the overall system can expect 5% EER or less (over 95% TAR and under 5% FAR). We quantize the rate of the shaded area over the plotted region for comparison. A larger area indicates better reconciliation performance with a more range of BAR to potentially exhibit 5% or less EER. In both schemes, the burstier error results in a less shaded area due to a flatter slope of the success rate as previously mentioned. Compared to the CS-based scheme, the ECC-based reconciliation has a higher rate of shaded area with 71.3%, 58.1% and 45.8% for independent, and Simple Gilbert models with $r = 0.2$ and 0.1, respectively.

These plots can be implemented as a Supported BAR model in our envisioned framework to output optimal reconciliation parameters. The black line, which represents the 50% success rate for varying parameters, is the guideline for selecting the optimal parameter. The parameter is chosen by taking the mean BAR between BAR_l and BAR_a, and selecting the corresponding BAR in the black line. If the given BAR_l and BAR_a are within (above and below) the shaded region, the target EER can be met. Otherwise, the target EER cannot be met and the authentication range must be decreased. While these plots only represent 5% EER (95% and 5% success rate line), further plots representing varying success rates can be pre-profiled within the Supported BAR model to verify the validity of the target EER.

4.2 Entropy loss

We next evaluate the quality of the final key, K, generated from B in terms of the resulting entropy. Note that the length of B is fixed to 128 bits ($b = 128$). In ECC-based reconciliation, the bit length of the final key, k is obtained by taking $b - 2T$ (assuming Reed-Solomon code). In CS-based scheme, k is equivalent to b because B is directly utilized as K. If only considering the key length, CS-based scheme retains more entropy. However, in CS-based scheme, when C_A is transferred over to Device B over a public channel, partial information is leaked to the potential adversary. Assuming a strong attack model where the adversary has access to a pre-established sensing matrix Φ, we have to assume that M bits out of k bits have been leaked. Considering this information leakage, we present the resulting entropy of K from the two reconciliation schemes in Figure 5. The parameters T and M are set to 12–38 and 41–96, respectively, representing the equivalent reconciliation performance derived from the previous analysis in Section 4.1. In both schemes, as T or M increases to tolerate higher error, the resulting entropy decreases linearly. Specifically, in the ECC-based scheme with $T = 12$, we can expect the final key to retain 104 bits of entropy, whereas with $T = 38$ that can tolerate as low as 70% BAR retains 52 bits of the entropy. Overall, the CS-based reconciliation experiences higher entropy loss within the same reconciliation performance range. When $M = 41$ to support 90% BAR, the resulting entropy of the final key is 87 bits. An increase in M up to 96 will result in K retaining only 32 bits. Considering the two equal reconciliation performance, ECC-based scheme results in retaining more bits of entropy in the final key.

4.3 Computation overhead

Finally, we compare the two schemes in terms of their computation costs. We implement the two schemes in C language on both client (Device A) and host (Device B) sides. As the main metrics, we use the *number of executed instructions* and the *execution time* measured using Linux `perf` command on the Raspberry Pi 4 equipped with an ARM Cortex-A72 1.4 GHz processor. The benchmark results of the schemes with varying T and M are illustrated in Figure 6(a) and (b), respectively (note the log scale on the y-axis). In both schemes, an increase in both T and M require a higher number of executed instructions due to the higher number of bit errors it can tolerate. In the ECC-based scheme, Device A merely performs an encoding function (involving linear operations to add redundancies) that results in execution of 180k to 320k (1.4 to 1.5 ms of execution time) instructions as T increases from 12 to 38. On the other hand, Device B performs more number of instructions, ranging from 300k to 800k (1.5 to 3.2 ms of execution time), due to the higher computation costs of the decoding function. Nevertheless, both sides can execute under 3.2 ms regardless of the increase in T.

The CS-based scheme, on the other hand, is orders-of-magnitude more computation-intensive. In Device A, $k \times M$ matrix multiplication using Φ results in a relatively higher number of executed instructions, ranging from 500k to 1,000k as M increases from 41 to 97, respectively. This results in an execution time of 1.5 to 2.7 ms. On the other hand, Device B requires orders of magnitude more number of instructions ranging from 128M to 363M (48 to 100 ms of execution time) due to the heavy computation of the l1 minimization algorithm to recover sparse vector.

Overall, ECC-based reconciliation is computationally lighter and faster than CS-based scheme for the same reconciliation performance. Also, in both schemes, the computation overhead of the host and the client is significantly different, especially in the CS-based scheme. This asymmetry must be considered when implementing a ZIPA technique on resource-constrained devices so that the device with more resource play the host role.

5 Discussion

Reconciliation scheme selection: Our analysis suggests that the ECC-based scheme outperforms the CS-based scheme in terms of its reconciliation success rate, entropy loss, and computation overhead (communication overhead difference between two schemes are negligible). More importantly, the final key generated from the ECC-based scheme is considered more secure because it is initially generated from the pseudo-random number generator that is less likely to be predictable by the adversary (i.e., equal probability of having 0s and 1s). In CS-based scheme, the final key is directly being used from the environmental bit sequences; these bit sequences are not always truly random and many prior works have rejected its randomness hypothesis [4, 7]. Considering this security property in addition to our provided benchmark results, ECC-based scheme is more suitable for our proposed framework.

Framework implementation: Most existing ZIPA works require consistent tuning of their reconciliation parameters due to the dynamic nature of the noise sources that heavily depend on the environment. For instance, in a prior work generating keys using ambient audio [12], a parameter that establishes the matching final key only within 2 m in a crowded canteen environment may exhibit further authentication range in a quiet office environment which may allow unwanted devices to authenticate with legitimate ones. To dynamically adjust the range, it is important that our proposed framework is embedded in many resource-limited IoT or wearable devices. The implementation of our framework requires minimal storage space and software-level modifications. Considering three bit error models, our framework requires less than 2.5 kB of storage space to support up to 10% target EER validation when implemented using the ECC-based scheme. Using CS-based scheme, the storage requirement is around 5 kB, which can easily be supported even by today's resource-constrained devices.

6 Related Works

Researchers have explored various random environmental contexts as a source of entropy to generate random keys for ZIPA. For general IoT devices, examples include wireless signal strength [9], ambient sound [11], luminosity [10], powerline noise [4], inter-event timing [2], vehicle vibration [5], or combinations of them. ZIPA techniques within the wearable device domain aim to prove the co-presence of the devices worn by the same user. For instance, physiological and behavioral signatures from electrocardiography and body gestures have been proposed to simplify pairing and authentication procedures [6–8]. While these prior works successfully propose new sensing modalities, many works do not explicitly state the definition of co-location or provide the parameter values for different authentication ranges, which often leave the users to determine the right values based on different deployment environments. Further, the lack of comprehensive comparison of different key reconciliation schemes leads to different groups of work using different schemes, which hinders the coherence and fair comparison of many ZIPA techniques.

7 Conclusion

As a first step towards developing a systematic ZIPA pipeline, we proposed a novel framework to determine the proper reconciliation parameter for the desired authentication range. We analyzed the two commonly used reconciliation schemes in terms of their security, usability, and computation overhead to find a more suitable scheme for optimizing ZIPA operation. The proposed framework can be embedded in many resource-constrained devices to dynamically adjust its authentication range and adapt based on its deployed environment.

Acknowledgement

This work was supported by the National Science Foundation under award CNS-1845469.

References

[1] E. N. Gilbert. 1960. Capacity of a burst-noise channel. *The Bell System Technical Journal* 39, 5 (1960).
[2] Jun Han et al. 2018. Do You Feel What I Hear? Enabling Autonomous IoT Device Pairing Using Different Sensor Types. In *IEEE Symposium on Security and Privacy (S&P)*.
[3] Ari Juels and Martin Wattenberg. 1999. A Fuzzy Commitment Scheme. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*.
[4] Kyuin Lee et al. 2019. VoltKey: Continuous Secret Key Generation Based on Power Line Noise for Zero-Involvement Pairing and Authentication. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 3 (2019).
[5] Kyuin Lee et al. 2020. ivPair: Context-Based Fast Intra-Vehicle Device Pairing for Secure Wireless Connectivity. In *ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*.
[6] Xiaopeng Li et al. 2020. T2Pair: Secure and Usable Pairing for Heterogeneous IoT Devices. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*.
[7] Qi Lin et al. 2019. H2B: Heartbeat-based Secret Key Generation Using Piezo Vibration Sensors. In *International Conference on Information Processing in Sensor Networks (IPSN)*.
[8] Qi Lin et al. 2020. KEHKey: Kinetic Energy Harvester-Based Authentication and Key Generation for Body Area Network. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020).
[9] Suhas Mathur et al. 2011. ProxiMate: Proximity-based Secure Pairing Using Ambient Wireless Signals. In *International Conference on Mobile Systems, Applications, and Services (MobiSys)*.
[10] Markus Miettinen et al. 2014. Context-Based Zero-Interaction Pairing and Key Evolution for Advanced Personal Devices. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*.
[11] Markus Miettinen et al. 2018. Revisiting Context-based Authentication in IoT. In *Design Automation Conference (DAC)*.
[12] Dominik Schürmann and Stephan Sigg. 2013. Secure Communication Based on Ambient Audio. *IEEE Transactions on Mobile Computing* 12, 2 (2013).
[13] Verizon. 2016. *2016 Data Breach Investigations Report.* Technical Report.
[14] Yuezhong Wu et al. 2020. Auto-Key: Using Autoencoder to Speed Up Gait-Based Key Generation in Body Area Networks. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020).
[15] Weitao Xu et al. 2019. LoRa-Key: Secure Key Generation System for LoRa-Based Network. *IEEE Internet of Things Journal* 6, 4 (2019).