DOI: 10.2478/ias-2025-0009 • JIAS • 20(3) • 2025 • 146-158

A Novel Forward-Secure Key Establishment Protocol from Physically Related Functions

Lakshmi Kuppusamy[1], B R Shankar[2], and Cheng-Chi Lee[3,4,*]

1 Department of Mathematics, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Udupi, 576104, India; k.lakshmidevi@manipal.edu
2 Department of Mathematical and Computational Sciences, National Institute of Technology, Surathkal, Karnataka, 575025, India; shankarbr@nitk.edu.in
3 Department of Library and Information Science, Fu Jen Catholic University, New Taipei City, 24206, Taiwan; cclee@mail.fju.edu.tw
4 Department of Computer Science and Information Engineering, Asia University, Taichung,41354, Taiwan
*Corresponding author: cclee@mail.fju.edu.tw

Abstract: Physically Related Functions (PReFs) are fundamentally similar to strong Physically Unclonable Functions (PUFs). Their study was driven by the goal of realizing ``Cryptophasia in hardware''—a phenomenon where two hardware devices can securely establish an authenticated key without the need for long-term cryptographic key storage. Chatterjee *et al.* demonstrated that devices using PReFs can achieve Authenticated Key Exchange (AKE) without the burden of intensive cryptographic computations, making this approach desirable for lightweight environments. However, to ensure forward secrecy, they employed an elliptic curve Diffie-Hellman protocol, with PReFs replacing long-term keys typically used in traditional public key-based AKE systems. This paper proposes a method to eliminate dependence on conventional public key cryptography for achieving forward-secure AKE under minimal and practical assumptions. Our forward-secure AKE protocol is constructed using PReFs, error-correction codes, and collision-resistant hash functions, and is proven secure under CK and eCK security models.

Keywords: PUF, PReF, AKE, forward secrecy, resource consumption

1 Introduction

Authenticated Key Exchange (AKE) protocols enable two parties to establish secure communication over a public network. The main security objective of an AKE protocol is to make it computationally infeasible for an adversary eavesdropping on the communication to learn the session secret shared between the parties. Traditionally, public key cryptography is employed to design forward-secure AKE protocols. However, these methods often require computationally intensive operations and long-term cryptographic keys [1, 2], which pose challenges in resource-constrained environments. In contrast, symmetric key-based primitives are more suitable for such devices, but managing keys in these scenarios remains a significant issue. *Cryptophasia in hardware* refers to the phenomenon where two hardware devices can achieve authenticated key exchange. Accomplishing this without relying on costly cryptographic techniques would be highly beneficial for resource-limited devices. Physical protection techniques [3] can be adopted for the resource constrained devices to implement symmetric key based primitives for realizing secure and efficient communication. However the large scale adoption of such techniques is difficult due to cost and scalability issues [4, 5]. In addition, recent works [4, 6] exposed attacks [7] on the secure key management techniques implemented in non-volatile memories such as RAM.

Motivated by these issues, Chatterjee *et al.* [8, 9] proposed physically related functions (PReFs), a specialized hardware component realized using strong Physically Unclonable Functions (PUFs). Their primary goal is to design a secure AKE protocol relying solely on PReFs and hash functions, which was seen as impossible without conventional cryptography [10]. Their proposal embeds PReFs during device manufacturing, which allows the devices implement AKE protocol with the input/output pairs generated by the PReFs. In their basic AKE protocol [9], the session key is masked by the output of a

PReF function embedded in one party's device, which is then transmitted to the other party. Since PReFs produce different outputs for the same input by design, the protocol uses Error Correcting Codes (ECC) [11] to reconcile the output differences between the devices and recover the correct session key. Further, they enhanced the PReF-based protocol to achieve forward secrecy by incorporating the elliptic curve-based Diffie-Hellman. *From this, we observe that while PReFs can effectively replace long-term keys in traditional AKE-based key exchange protocols without compromising security, still public-key method was used to achieve forward security.*

In this paper, we aim to fully eliminate traditional public key based approaches from the protocol proposed in [9]. In particular, we make the necessary changes to the design of Chaterjee *et al.* and present a forward-secure AKE protocol that primarily depends on PReFs, removing the need for both public key operations and long-term cryptographic keys. We show that our protocol is secure under the security models of [1, 2]. It is evident from Table 1 and Table 2 that our protocol in Figure 2 is better suited for resource constrained devices.

Outline. In Section 2, our AKE protocol is presented along with the security analysis. Section 3 presents our proposal for dynamic networks and Section 4 concludes the paper. To ease readers, Supplementary A provides a brief introduction to PReFs and relevant definitions from [9] and a complete description of Chatterjee *et al.* protocol is reproduced in Section B.

2 A Forward Secure AKE Protocol from PReFs

Physically Related Functions introduced by Chatterjee *et al.* has the potential to become a standalone primitive in the design of key exchange protocols [8, 9]. Their PReF-based AKE protocol avoids secure storage by replacing PReFs for long-term keys (widely used to achieve authentication in key exchange protocols). However, it remains a challenging problem to design a AKE protocol that is forward secure and relies only on PReFs, error correction and collision resistant hash functions. Inspired by the approach of Chatterjee *et al.*, we propose a PReF based AKE that is forward secure and does not require any public key cryptography techniques to achieve forward secrecy.

System model of the protocol for Static PReF Network. Let $\mathscr{F} = \{f_i : \mathscr{X} \to \mathscr{Y}\}$ be a function family, where $\mathscr{X} = \{0,1\}^m$ and $\mathscr{Y} = \{0,1\}^n$ are arbitrary sets. Let Alice and Bob are two parties who physically implement the functions f_1 and f_2 using their respective hardware devices D_1 and D_2. There is a trusted (semi-honest) party, such as a manufacturer, which performs the following actions before deploying devices D_i (for $i = 1,\ldots,T$) in the static PReF network:

(a) A $(\mathsf{T}, \varepsilon, \varepsilon', \delta)$-ReF graph is constructed, represented by the functions $(f_1, f_2, \ldots, f_{\mathsf{T}}) \in \mathscr{F}$, where each function f_i is embedded in the corresponding device D_i (for $i = 1,\ldots,T$).

(b) For every pair of devices D_i and D_j, the party who is assumed to be trusted publish the related input set $\mathscr{X}_{i,j}$ on a cloud/ledger which is publicly accessible.

Alice and Bob (devices D_1 and D_2) agree on an input x from the publicly available related input set $\mathscr{X}_{1,2}$ and compute the output y using their respective PReFs, f_1 and f_2. An adversary with access to f_0 cannot predict the output y, even if they have access to the PReF pairs $\mathscr{X}_{1,0}$ and $\mathscr{X}_{2,0}$, because

$$\mathscr{X}_{1,2} \cap \mathscr{X}_{1,0} = \mathscr{X}_{1,2} \cap \mathscr{X}_{2,0} = \emptyset.$$

We begin with a proposal of one-pass key exchange protocol based on PReF as illustrated in Figure 1. This protocol relies solely on the existence of the PReF network and the reconciliation of related outputs. To mitigate security vulnerabilities highlighted in [12], the protocol employs BCH-based error correction to reconcile the outputs of PReFs, and subsequently establishes the session key, as outlined in [9]. As depicted in Figure 1, two parties, $\mathscr{D}_1$ and $\mathscr{D}_2$ establish a session key upon executing the protocol. $\mathscr{D}_1$ initiates the session as follows: $\mathscr{D}_1$ randomly selects an input x from the input set $\mathscr{X}_{1,2}$ and computes y_1 using the function f_1. Then, $\mathscr{D}_1$ computes A based on y_1, a random value r, and the session-specific secret $k_{1,2}$. The values A and x are sent to $\mathscr{D}_2$. Upon receiving A and x, $\mathscr{D}_2$ computes y_2 using $f_2(x)$ and then decodes A using y_2 and $k_{1,2}$ to retrieve r. Both devices then agree on r as the session key.

To ensure protection against replay, impersonation, and integrity attacks, a collision-resistant hash function $\mathscr{H} : \{0,1\}^* \to \{0,1\}^n$ and a session-specific nonce TS are introduced. Thus, an attacker finds it computationally infeasible to find $\mathscr{H}(r||\mathsf{TS}||\mathscr{D}_1||\mathscr{D}_2) = \mathscr{H}(r'||\mathsf{TS}'||\mathscr{D}_1||\mathscr{D}_2)$ for a given (r, TS) such that values $r' \neq r$ and $TS' \neq TS$. The resultant AKE protocol after incorporating the above measures is depicted in Fig. 2. The session key r remains secure even if an adversary gains access to the output of the PReF functions f_1 or f_2 for the input x. This is because the adversary is unable to derive r from A and y_1

$\mathscr{D}_1$		$\mathscr{D}_2$
session secret $k_{1,2} \in \{0,1\}^n$		session secret $k_{1,2} \in \{0,1\}^n$
select a random $x \in \mathscr{X}_{1,2}$		
random $r \in \{0,1\}^n$		
Compute $y_1 = f_1(x)$		
$A = E(r) + y_1 + k_{1,2}$	$\xrightarrow{x, A}$	Compute $y_2 = f_2(x)$
		$r' = D(A + y_2 + k_{1,2})$
session key $\mathsf{SK} = r = r'$.		
session-specific secret update for the next session $k_{1,2} \leftarrow H(r, k_{1,2})$		

Figure 1. Our Basic Key Exchange protocol using PReF and ECC

$\mathscr{D}_1$		$\mathscr{D}_2$
session secret $k_{1,2} \in \{0,1\}^n$		session secret $k_{1,2} \in \{0,1\}^n$
nonce TS		
select a random $x \in \mathscr{X}_{1,2}$		
random $r \in \{0,1\}^n$		
Compute $y_1 = f_1(x)$		
$B = \mathscr{H}(r \Vert \mathsf{TS} \Vert \mathscr{D}_1 \Vert \mathscr{D}_2 \Vert x)$		
$A = E(r) + y_1 + k_{1,2}$	$\xrightarrow{x, A, B, \mathsf{TS}}$	Compute $y_2 = f_2(x)$
		$r' = D(A + y_2 + k_{1,2})$
		Check if $B = \mathscr{H}(r' \Vert \mathsf{TS} \Vert \mathscr{D}_1 \Vert \mathscr{D}_2 \Vert x)$
	$\xleftarrow{C}$	computes $C = \mathscr{H}(r' \Vert \mathsf{TS} \Vert \mathscr{D}_2 \Vert \mathscr{D}_1 \Vert x)$
Check if $C = \mathscr{H}(r \Vert \mathsf{TS} \Vert \mathscr{D}_2 \Vert \mathscr{D}_1 \Vert x)$		
session key $\mathsf{SK} = r = r'$.		
session-specific secret update for the next session $k_{1,2} \leftarrow H(r, k_{1,2})$		

Figure 2. Our Forward Secure AKE protocol using PReFs, ECC and Hash Function

or y_2, as the ephemeral secret $k_{1,2}$ remains unknown to them. Additionally, using $k_{1,2}$ offers the benefit of allowing x to be reused across multiple authentications. The adversary cannot deduce $r + s$ from two sessions that using same x but different r and s.

Remark. Our protocol differs from the one proposed by Chatterjee *et al.* (refer to Figure 6) by utilizing a session-specific secret $k_{1,2}$ between the devices $\mathscr{D}_1$ and $\mathscr{D}_2$, thereby eliminating the need for the elliptic curve Diffie-Hellman protocol. Specifically, we assume that $k_{1,2}$ is a pre-shared short-term secret stored in the devices during deployment. When a session is initiated between $\mathscr{D}_1$ and $\mathscr{D}_2$, this secret $k_{1,2}$ is used to establish the session key. Afterward, this short-term secret is updated using a collision-resistant hash function H, as depicted in Figure 2. This ensures that $k_{1,2}$ acts as an ephemeral secret specific to each session.

2.1 Security Analysis

In this section, we show that our protocol is forward secure according to the standard AKE models, namely Canetti-Krawczyk model [2] and extended Canetti-Krawczyk model [1].

AKE Model for PReF Network. The model assumes that all devices $\mathscr{D}_1, \ldots, \mathscr{D}_\mathsf{T}$ and the adversary $\mathscr{A}$ run as Probabilistic Polynomial Time (PPT) algorithms. The adversary $\mathscr{A}$ is capable of initiating multiple instances of the protocol between any pair of devices $(\mathscr{D}_i, \mathscr{D}_j)$, with each instance identified as a *session*. Every session is attributed by a unique nonce or session identifier TS, preventing the adversary from reusing the same nonce to start multiple sessions. During the experiment, $\mathscr{A}$ can issue the following oracle queries:

Key Reveal (TS): This query outputs the session key and nonce TS

Output Reveal ($x \in \mathscr{X}'_{i,j}$): The output of the functions embedded in $\mathscr{D}_i$ or $\mathscr{D}_j$ is revealed through this query. Whenever an adversary issues this query with input x, the respective devices that reveals the

corresponding long-term secret y are said to be corrupt. Note that this query is analogous to the long-term key reveal query defined in the eCk model [1].

Ephemeral Key Reveal (TS): This query outputs ephemeral key of the session identified by TS.

Test (TS): This query is made only once during the entire experiment. Whenever this query is issued, the challenger chooses a bit $b \in \{0,1\}$ and reveals either the session key of the session with TS or a uniformly random key.

The adversary uses the above queries to establish a second session with the same key as the targeted session. The adversary is permitted to issue the test query only on a fresh session that meets the following conditions:

- $\mathscr{A}$ does not issue Key Reveal query for the test session.
- Whenever $\mathscr{A}$ issues ephemeral key reveal query, then it is not allowed to issue output reveal query for the corresponding pair of devices.
- Whenever $\mathscr{A}$ issues output reveal query for the devices, then it is not allowed to issue ephemeral key reveal query for the same pair of devices.

After issuing a test query on the fresh session, the adversary's goal is to distinguish the session key from a random key. Consequently, the adversary wins the experiment only when it distinguishes the session key from a random key with a non-negligible probability.

Theorem. The protocol in Figure 2 is a forward secure AKE protocol provided: (a) the pair of functions (f_i, f_j) embedded in the respective devices $\mathscr{D}_i$ and $\mathscr{D}_j$ that is associated with a $(T, \varepsilon, \varepsilon', \delta)-$PReF network are pseudorandom by Definition A.1, (b) r is chosen uniformly at random and (c) $\mathscr{H}$ is modelled as a random oracle.

Proof. We first show that the functions f_i and f_j are pseudorandom. Then we argue the security of the our protocol assuming the pseudorandomness of the functions f_i and f_j.

Pseudorandomness. Let $\mathscr{C}$ represents a PPT algorithm embedded with the function f_i. $\mathscr{C}$ selects $b \in \{0,1\}$ at random and computes $y_i = E(r) + y_i + k$, for a PReF input $x \in \mathscr{X}'_{i,j}$ and a random $r \in \{0,1\}^n$. Whenever b is 0, it sets $y_i = f_i(x)$ and $k = k_{1,2}$. Otherwise, it sets $y_i = s$ and $k = k'$ where s and k' are chosen at random from $\{0,1\}^n$.

Let $\varepsilon(\lambda)$ be a non-negligible function and $\mathscr{D}$ be the PPT distinguisher having non-negligible advantage of distinguishing the output $E(r) + f_i(x) + k_{1,2}$ from $E(r) + s + k'$. That is, for all $r \in \{0,1\}^n$ and $s \in \{0,1\}^n$, the advantage of $\mathscr{D}$ is

$$| \Pr[\mathscr{D}(E(r) + f_i(x) + k_{1,2}) = 1] - Pr[\mathscr{D}(E(r) + s + k') = 1] | = \varepsilon(\lambda).$$

Now we show that there exist a PPT algorithm $\mathscr{A}$ against the pseudorandomness of the function f_i that uses $\mathscr{D}$ to win the pseudorandom experiment against the challenger as follows:

- Whenever an input 1^λ is received from the challenger, the adversary $\mathscr{A}$ submits $x \in \mathscr{X}'_{i,j}$
- Upon receiving the response x from $\mathscr{A}$, the challenger selects $b \in \{0,1\}$ and assigns $y_i = f_i(x)$ and $k = k_{1,2}$ if $b = 0$. Otherwise it sets $y_i = s$ and $k = k'$ for a random s and k' chosen from $\{0,1\}^n$.
- After receiving y_i from the challenger, $\mathscr{A}$ simulates the algorithm $\mathscr{C}$ to distinguisher $\mathscr{D}$. Finally $\mathscr{A}$ sets the output $b' = 0$ if $\mathscr{D}$ outputs 0. Otherwise b' is set to 1.

Clearly the adversary $\mathscr{A}$ wins the challenger with a non-negligible advantage $\varepsilon(\lambda)$ as the distinguisher $\mathscr{D}$ wins the challenge with the advantage $\varepsilon(\lambda)$ which is a contradiction to the definition of Pseudorandomnes of f_i. Thus any distinguisher against $\mathscr{C}$ can distinguish $E(r) + f_i(x) + k_{1,2}$ from $E(r) + s + k'$ only with negligible advantage.

AKE Proof. Assume that there exists a PPT adversary $\mathscr{A}$ against our AKE protocol in Figure 2. In the AKE experiment, the aim of an adversary is to distinguish between the session key of the test session and a random key.

Consider an event E_1 where an adversary tries to establish a second session with the same session key computed in the target session and then wins the experiment by issuing a session key reveal query to the second session. We now analyze the probability of event E_1 occurring. According to the AKE experiment definition, the adversary is not allowed to use the same nonce to initiate multiple sessions TS, although they can choose the same input x for both sessions. Let the session key for the target session be r and for the second session be m. The only way the adversary can replace m with r is by determining the unknowns $k_{1,2}$ and y_i from A and making the checksum $B = \mathscr{H}(r||\mathsf{TS}||\mathscr{D}_1||\mathscr{D}_2||x)$ valid. However, this is negligible under the assumption that the hash function $\mathscr{H}$ is collision-resistant.

Recall that a test query is issued against a fresh session where the adversary is allowed to issue either the ephemeral key reveal query or the output reveal query. Now define an event E_2 where an adversary tries to distinguish the target session key from a random key after obtaining secret information used in the target session through the following queries:

Case 1. The adversary $\mathscr{A}$ issues only the ephemeral key reveal query for the session it targets. The adversary obtains the random value $k_{1,2}$ used in the session. The adversary having $k_{1,2}$ and x has to compute r from A which is sent public. $\mathscr{A}$ cannot distinguish $E(r) + f_i(x) + k_{1,2}$ from $E(r) + s + k_{1,2}$ for a random s as the PReF functions are pseudorandom. On the other hand, $\mathscr{A}$ cannot recover r from $B = \mathscr{H}(r||\mathsf{TS}||\mathscr{D}_1||\mathscr{D}_2||x)$ since $\mathscr{H}$ is modelled as a random oracle and r is chosen uniformly at random. From the above arguments, it is clear that $\mathscr{A}$ can forge the session only with negligible probability even if the ephemeral key is revealed.

Case 2. Now consider the case where an adversary $\mathscr{A}$ issues the output reveal query (x) on the target session to obtain $f_i(x)$. Then the adversary is not allowed by definition to use the same x for a new session. The adversary is unable find the session key r from A and $f_i(x)$ without knowing $k_{1,2}$. Thus the adversary is successful in forging the session key of the target session only with negligible probability. In particular, it is clear that the above argument holds when the adversary tries to compute session keys of the completed session. In the case of completed sessions, the adversary could not retrieve $k_{1,2}$ of the particular session as they are updated at the end of each session and hence the session key r. With this, we claim that our AKE protocol is forward secure. □

2.2 A comparative Analysis with the Existing (PUF based) AKE Protocols.

This section compares our protocol with existing (PUF based) AKE protocols in terms of computational cost and forward secrecy property. In particular, the computation cost is calculated by referring the work of Shahidinejad and Abawajy [13] where the authors reported the execution time of algorithms listed in Table 1 and Table 2 in both the Intel and ARM processors.

2.2.1 *Comparison With AKE protocols.*

The major goal of our protocol design is to completely avoid public key based cryptographic computations and to depend only on the existence of PReFs and collision resistant hash functions. To achieve this our protocol described in Section 2 uses a session-specific secret $k_{1,2}$ which is stored in the devices during deployment. When the first session is initiated between $\mathscr{D}_1$ and $\mathscr{D}_2$, the value $k_{1,2}$ is used to establish the first session key. In the subsequent sessions, the value $k_{1,2}$ gets updated as depicted in Figure 1. This methodology is proposed not only to avoid public key computations but also to ensure that $k_{1,2}$ acts as an ephemeral secret specific to each session. During deployment, secret key erasure techniques may be implemented to destruct/zeroizing the secret values [22] to make them inaccessible and avoid the attack of [7]. Our protocol thus avoids storage of long-term secret values when compared with the existing secure storage based AKE protocols [14, 15, 19, 23]. Thus it is evident from Table 1 that our protocol in Figure 2 is more efficient when compared with the other AKE protocols. Note that our PReF based AKE protocol achieves forward secrecy using four hash computations whereas the protocol proposed by Dousti *etal.* [18] requiring six hash computations lacks forward secrecy.

2.2.2 *Comparison With PUF Based AKE protocols.*

Many PUF based AKE protocols are proposed in the literature [24–30] for resource constrained environments as they are extremely light weight and unpredictable. However the use of PUFs introduces the need to securely store the secret values and to engage in expensive computations [27, 31]. In particular, a root-of-trust is established in protocols [24–26, 29] by a trusted third party that handles secure storage of the data involved and/or aids the light weight devices to accomplish authentication during the protocol run.

Table 1. Comparison: SK, EPA, PR, EPM and MEx refer to symmetric key operations, elliptic curve point addition, pairing on elliptic curves, elliptic curve point multiplication and the number of exchanged messages respectively

Scheme	Computation Time in μs			Forward secrecy	Comm. overhead (bit)	storage
	Operations	Intel	ARM			
[14]	11 hash	≈ 3.41	≈ 49.55	yes	1088	yes
[15]	8 EPM & 6EPA & 2 hash	≈ 602.48	≈ 6830.25	no	4160	yes
[16]	3 RSA & 2 hash	≈ 1146.62	≈ 23283.01	yes	1536	yes
[17]	4 SK & 5 hash	≈ 1.95	≈ 25.905	yes	512	yes
[18]	6 hash	≈ 1.86	≈ 27.03	no	1280	yes
[19]	6 EPM & 2 EPA & 6 hash	≈ 452.48	≈ 5134.11	yes	1088	yes
[20]	3 RSA & 19 hash & 2 signature	≈ 578.87	≈ 11722.61	yes	6208	yes
[21]	10 EPM & 2 EPA & 2 PR & 4 hash	≈ 2207.86	≈ 25015	yes	2048	yes
[9]	4 EPM & 6 hash	≈ 101.86	≈ 1627.03	yes	1216	no
Ours	4 hash	≈ 1.24	≈ 18.02	yes	704	no

On the other hand, protocols that eliminate the need of trusted third party either requires expensive elliptic curve based computations [26, 28] or lacks stronger security such as forward secrecy [27]. In a recent article, Boyapally *et al.* [30] introduced the concept of Harmonizing PUFs (H PUF) and proposed a AKE protocol based on H PUF to achieve efficiency in terms of communication overhead, computation cost and security that depend only on the H PUF and Hash functions as in Table 2. Our PReF based protocol achieves stronger security under the assumption on PReF and Hash functions with minimal computational and communication cost.

3 PReF based AKE for Dynamic Networks

This section proposes a forward secure PReF based AKE protocol for a dynamic PReF network. The approach is similar to the one proposed in [9] where a designated set of honest devices (referred as auxiliary devices) are employed (a) to introduce new devices to the network and (b) to assist the existing devices that could not communicate with the other devices in the network due to the compromise of PReF to re-join the network. The dynamic network consists of two types of devices: (1) primary devices that take part in the key exchange protocol for establishing secure communication with other devices and (2) auxiliary devices that are **honest-but-curious** and does not take part in the key exchange process with primary devices. Auxiliary devices are solely responsible for assisting new (or primary) devices in joining or rejoining the network.

Protocol Description: Let T and S be the number of primary and auxiliary devices respectively in the network . The major goal of the auxiliary device is to enable the devices (without pre-established PReF pairs with any of the devices in the network) to establish a session key between the primary devices. For a a family of functions $\mathscr{F} = \{f_i : \mathscr{X} \rightarrow \mathscr{Y})\}$, where $\mathscr{X} = \{0,1\}^m$ and $\mathscr{Y} = \{0,1\}^n$ are arbitrary sets. Let $f_i, i = 1 \ldots T+S$ be a $(\mathsf{T}+\mathsf{S}, \varepsilon, \varepsilon^{'}, \delta)-$ReF graph represented by the functions $(f_1, f_2, \ldots f_{\mathsf{T}+\mathsf{S}}) \in \mathscr{F}$ is constructed where each f_i is embodied to the corresponding devices $D_i, i = 1, \ldots T+S$.

As illustrated in Figure 3, $\mathscr{D}_0$ is the auxiliary device that enables D_2 (without the unique set $\mathscr{X}_{1,2}$) to establish a session key with the primary device D_1 by communicating with D_1 using the related input set $\mathscr{X}_{0,1}$ and the preshared secret $k_{0,1}$. Note that the session key derived between D_1 and D_2 is known to the auxillary device D_0 which is assumed to be honest in our protocol description. Our protocol enables the addition of new devices with the computation of: (1) two error correction, two exclusive OR and four hash operations for each of the primary devices and (2) two error correction and five hash computations for the auxiliary device. The following theorem shows that our protocol is weak perfect forward secure.

Theorem. The protocol in Figure 3 is a forward secure AKE protocol provided: (a) the pair of functions (f_i, f_j) embedded in the respective devices $\mathscr{D}_i$ and $\mathscr{D}_j$ that is associated with a $(T, \varepsilon, \varepsilon', \delta)-$PReF network are pseudorandom by Definition A.1, (b) r is chosen uniformly at random and (c) $\mathscr{H}$ is modelled

Table 2. Comparison: SK, EPA, PR, EPM and MEx refer to symmetric key operations, elliptic curve point addition, pairing on elliptic curves, elliptic curve point multiplication and the number of exchanged messages respectively

Scheme	Computation Time in μs			Forward secrecy	Communication overhead (in bits)	storage
	Operations	Intel	ARM			
[24]	4 hash & 12 MAC & 10 ENC	≈ 7.28	≈ 78.07	yes	7872	yes
[25]	8 Hash & 6 hash-to-point & 4 EPM	≈ 304.34	≈ 3463.07	yes	9312	yes
[26]	14 Hash & 20 hash-to-point & 8 EPM & 16 EPA & 4 PR	≈ 3527.5	≈ 39989.8	yes	9856	yes
[27]	6 Hash & 12 PRF & 3 GroupExp	≈ 579.9	≈ 11715.6	yes	1952	yes
[28]	12 Hash & 14 EPM & 4 EPA	≈ 1054.96	≈ 11968.2	yes	4160	yes
[29]	12 Hash & 4 Enc	≈ 4.12	≈ 57.44	yes	2880	yes
[30]	8 hash	≈ 2.48	≈ 36.04	yes	1216	yes
Ours	4 hash	≈ 1.24	≈ 18.02	yes	704	no

$\mathscr{D}_1$		$\mathscr{D}_0$		$\mathscr{D}_2$
session secret $k_{0,1} \in \{0,1\}^n$		session secret $k_{0,j} \in \{0,1\}^n, j = 1,2$		session secret $k_{0,2} \in \{0,1\}^n$
nonce TS_1				nonce TS_2
select a random $x_1 \in \mathscr{X}_{0,1}$				select a random $x_2 \in \mathscr{X}_{0,2}$
random $r_1 \in \{0,1\}^n$				random $r_2 \in \{0,1\}^n$
Compute $y_1^{0,1} = f_1(x_1)$				Compute $y_1^{0,2} = f_2(x_2)$
$B_1^{0,1} = \mathscr{H}(r_1 \| \mathsf{TS}_1 \| \mathscr{D}_0 \| \mathscr{D}_1 \| x_1)$				$B_1^{0,2} = \mathscr{H}(r_2 \| \mathsf{TS}_2 \| \mathscr{D}_0 \| \mathscr{D}_2 \| x_2)$
$A_1^{0,1} = E(r_1) + y_1^{0,1} + k_{0,1}$	$\xrightarrow{x_1, A_1^{0,1}, B_1^{0,1}, \mathsf{TS}_1}$		$\xleftarrow{x_2, A_1^{0,2}, B_1^{0,2}, \mathsf{TS}_2}$	$A_1^{0,2} = E(r_2) + y_1^{0,2} + k_{0,2}$
		$y_2^{0,1} = f_0(x_1)$		
		$y_2^{0,2} = f_0(x_2)$		
		$r_1^{'} = D(A_1^{0,1} + y_2^{0,1} + k_{0,1})$		
		$r_2^{'} = D(A_1^{0,2} + y_2^{0,2} + k_{0,2})$		
		$B_1^{0,1} \stackrel{?}{=} \mathscr{H}(r_1^{'} \| \mathsf{TS}_1 \| \mathscr{D}_0 \| \mathscr{D}_1 \| x_1)$		
		$B_1^{0,2} \stackrel{?}{=} \mathscr{H}(r_2^{'} \| \mathsf{TS}_2 \| \mathscr{D}_0 \| \mathscr{D}_2 \| x_2)$		
		select a random $x_3 \in \mathscr{X}_{0,1}$		
		select a random $x_4 \in \mathscr{X}_{0,2}$		
		Compute $y_3^{0,1} = f_0(x_3)$		
		Compute $y_3^{0,2} = f_0(x_4)$		
		$C_1^{0,1} = \mathscr{H}(r_2^{'} \| \mathsf{TS}_1 \| \mathscr{D}_0 \| \mathscr{D}_1 \| x_3)$		
		$C_1^{0,2} = \mathscr{H}(r_1^{'} \| \mathsf{TS}_2 \| \mathscr{D}_0 \| \mathscr{D}_2 \| x_4)$		
		$A_2^{0,1} = E(r_2^{'}) + y_3^{0,1} + k_{0,1}$		
	$\xleftarrow{x_3, A_2^{0,1}, C_1^{0,1}, \mathsf{TS}_1}$	$A_2^{0,2} = E(r_1^{'}) + y_3^{0,2} + k_{0,2}$	$\xrightarrow{x_4, A_2^{0,2}, C_1^{0,2}, \mathsf{TS}_2}$	
Compute $y_4^{0,1} = f_1(x_3)$				Compute $y_4^{0,2} = f_2(x_4)$
$r_2^{''} = D(A_2^{0,1} + y_4^{0,1} + k_{0,1})$				$r_1^{''} = D(A_2^{0,2} + y_4^{0,2} + k_{0,2})$
$C_1^{0,1} \stackrel{?}{=} \mathscr{H}(r_2^{''} \| \mathsf{TS}_1 \| \mathscr{D}_0 \| \mathscr{D}_1 \| x_3)$				$C_1^{0,2} \stackrel{?}{=} \mathscr{H}(r_1^{''} \| \mathsf{TS}_2 \| \mathscr{D}_0 \| \mathscr{D}_2 \| x_4)$
session key $\mathsf{SK} = H(r_1, r_2)$.				
session secret $k_{0,j} = H(r_1, r_2, k_{0,j})$, $j = 1,2$				

Figure 3. A Forward-secure AKE protocol in Dynamic PReF Network

as a random oracle. In addition, the auxiliary device $\mathscr{D}_0$ is assumed to be honest and knows the session key established between the primary and new device.

Proof. The following AKE proof is given assuming the pseudorandomness of the functions f_i and f_j as shown in Theorem 2.1.

AKE Proof. Assume that there exists a PPT adversary $\mathscr{A}$ against our AKE protocol in Figure 3. In the AKE experiment, the aim of an adversary is to distinguish between the session key of the test session and a random key.

Consider an event E_1 where an adversary tries to establish a second session with the same session key computed in the target session and then wins the experiment by issuing a session key reveal query to the second session. We now analyze the probability of event E_1 occurring. According to the AKE experiment definition, the adversary is not allowed to use the same nonce to initiate multiple sessions TS,

although they can choose the same input x for both sessions. Let the session key for the target session be r and for the second session be m. The only way the adversary can replace m with r is by determining the unknowns $k_{1,2}$ and y_i from A and making the checksum $B = \mathscr{H}(r||\mathsf{TS}||\mathscr{D}_1||\mathscr{D}_2||x)$ valid. However, this is negligible under the assumption that the hash function $\mathscr{H}$ is collision-resistant.

Recall that a test query is issued against a fresh session where the adversary is allowed to issue either the ephemeral key reveal query or the output reveal query. Now define an event E_2 where an adversary tries to distinguish the target session key from a random key after obtaining secret information used in the target session through the following queries:

Case 1. The adversary $\mathscr{A}$ issues only the ephemeral key reveal query for the session it targets. The adversary obtains the value $k_{0,1}$ and $k_{0,2}$ used in the session. The adversary having $k_{0,1}$, $k_{0,2}$, x_1 and x_2 has to compute r_1 and r_2 from $A_1^{0,1}$ or $A_2^{0,2}$ and $A_1^{0,2}$ or $A_2^{0,1}$ respectively which are sent public. $\mathscr{A}$ cannot distinguish $E(r_1)+y_1^{0,1}+k_{0,1}$ from $E(r_1)+s+k_{0,1}$ for a random s as the PReF functions are pseudorandom. On the other hand, $\mathscr{A}$ cannot recover r_1 from $B_1^{0,1} = \mathscr{H}(r_1||\mathsf{TS}_1||\mathscr{D}_0||\mathscr{D}_1||x_1)$ since $\mathscr{H}$ is modelled as a random oracle and r_1 is chosen uniformly at random. The similar arguement holds for recovery of r_2. Thus, it is clear that $\mathscr{A}$ can forge the session only with negligible probability even if the ephemeral key is revealed.

Case 2. Now consider the case where an adversary $\mathscr{A}$ issues the output reveal query $(x_j,)\, j = 1\ldots4$ on the target session to obtain $f_i(x_j), i = 1,2$ and $j = 1\ldots4$. Then the adversary is not allowed by definition to use the same $x_j, j = 1\ldots4$ for a new session. The adversary is unable to find the keys r_1 and r_2 from $A_1^{0,1}$ or $A_2^{0,2}$ and $A_1^{0,2}$ or $A_2^{0,1}$ respectively and $f_i(x_j)$ without knowing $k_{0,1}$ and $k_{0,2}$. Thus the adversary is successful in forging the session key of the target session only with negligible probability. In particular, it is clear that the above argument holds when the adversary tries to compute session keys of the completed session. In the case of completed sessions, the adversary could not retrieve $k_{0,1}$ and $k_{0,2}$ of the particular session as they are updated at the end of each session and hence the session key SK. With this, we claim that our AKE protocol desribed in Figure 3 is weak perfect forward secure. □

4 Conclusion

Cryptophasia is a phenomenon which enables twins to communicate in a language that only they can understand. Chatterjee *et al.* investigated how to achieve ``Cryptophasia in Hardware'' so that a device pair can securely communicate with each other over an insecure/open network. Chatterjee *et al.* stated, ``*what if the participating devices are far too resource- constrained to either support the heavy computational costs of such a key exchange protocol or the secure storage requirements to store the resulting secret key? In particular, constraints with respect to computational resources make it dificult to deploy standard public-key techniques such as public-key encryption and or digital signatures to achieve secure and authenticated communication.*'' In this work, we showed that it is feasible to obtain forward-secure AKE protocol without assuming trusted party and public-key crypto computations for the static network. Our forward secure AKE protocol for the static network is built using PReFs, error-correction codes and collision resistant hash functions and is shown to be secure under the widely-accepted security model for AKE protocols. In case of dynamic network, we could achieve it by assuming that the auxiliary devices are trusted and hence designing a PReF based AKE protocol without the assumption of trusted auxiliary devices and random oracles is left for future work.

A Physically Related Functions and its Realisation in Hardware

This section intends to describe the main difference between PReFs and PUFs as the former requires additional assumptions to be realised in hardware. The main objective of PReF is to set randomised functionalities in hardware [8] similar to strong PUF. Contrary to the design of PUF where the similarity in responses of a PUF pair for an entire input space is expected to be unpredictable/uncorrelated, PReF is defined to exhibit its usage on correlated responses for specific values in the input space. This in turn, makes PReFs a predominant function to design efficient AKE protocols as PUF-based AKE protocols need additional assumptions such as the existence of secure channels and/or a trusted third party. On the other hand, it is more challenging to design PReFs compared to PUFs as they rely on :(a) internal devices knowledge to generate related input/output pairs during the setup phase and (b) avoiding the exposure of related input/output pairs to an adversary only having black-box access to the devices [9].

A.1 Notations and Definitions

Denote the security parameter by $\lambda \in \mathbb{N}$. The following definitions are reproduced from [9].

Ideal PReF pairs. Let f_A and f_B be the pair of functions with input and output space $\mathscr{X}$ and $\mathscr{Y}$ respectively. We call f_A and f_B be the related function pair whenever the output behaviour of f_A and f_B are *identical* for a specific input from $\mathscr{X}$. Those inputs form a subset $\mathscr{X}_{A,B} \subseteq \mathscr{X}$. Note that the output behaviours of f_A and f_B are uncorrelated for inputs outside of the subset $\mathscr{X}_{A,B}$. More formally, it should be difficult for a computationally bounded adversary to correctly guess the output of $f_B(x)$ and $f_A(x)$ for a given input $x \nsubseteq \mathscr{X}_{A,B}$ with a probability much better than the random guess.

To realise PReF pairs in the real world using hardware characteristics, the outputs of PReFs pairs are allowed to be *non-identical* but related via a distance function as in the following definition.

Related Function (ReF). For a family of functions $\mathscr{F} = \{f_i : \mathscr{X} \to \mathscr{Y}\}$ and arbitrary sets $\mathscr{X}$ and $\mathscr{Y}$, assume that there is a well-defined distance function **dist:** $\mathscr{Y} \times \mathscr{Y} \to \mathbb{N}$ over a pair of elements in $\mathscr{Y}$. Then, each pair $(f_i, f_j) \in \mathscr{F} \times \mathscr{F}$ is defined as $(\varepsilon, \delta)-$related for constants $\varepsilon \in (0,1]$ and $\delta \in \mathbb{N}$, whenever there exists a set of inputs $\mathscr{X}_{i,j} \subseteq \mathscr{X}$ that satisfies the following conditions:

- $|\mathscr{X}_{i,j}| \geq \varepsilon |\mathscr{X}|$
- For any input $x \in \mathscr{X}_{i,j}$, $\mathbf{dist}[f_i, f_j] \leq \delta$.

Reconciliation. The major goal of an AKE protocol is to enable the parties to establish a common secret. In order to design AKE using ReFs, reconciliation to a common secret from the respective parties outputs (evaluated on the same input) has to be achieved. That is, for each ReF pair (f_i , f_j) with related input x and an appropriate δ, there must exist an error correcting code scheme such that :

$$\mathbf{dist}[f_i(x), f_j(x)] \leq \delta \iff \mathsf{Decode}(\mathsf{f_i}(\mathsf{x})) = \mathsf{Decode}(\mathsf{f_j}(\mathsf{x}))$$

It is evident that such a reconciliation can be achieved easily using well established information-theoretic techniques such as error-correcting codes (ECC) [11].

Error-Correction Code (ECC). [9] Let (E, D) be the encoding and decoding algorithms of an (n, k, δ) linear error correction code. Then,

$$D(E(a) + e) = a; \quad E(a) + E(b) = E(a + b)$$

where $a, bE(a), E(b) \in \{0,1\}^k$ and $\textsc{Hamming weight}(e) \leq \delta$.

Pseudo-randomness of ReF. Let $f_i \in \mathscr{F}, 0 \leq i \leq n$ be the functions that are (ε, δ) related on the respective input sets $\mathscr{X}_i \subseteq \mathscr{X}, 1 \leq i \leq n$. Let $\mathscr{A}$ be a probabilistic polynomial time (PPT) algorithm with oracle access to $f_i, 0 \leq i \leq n$. For an input $x \in \mathscr{X}$ and $b \in \{0,1\}$, define the event $\mathsf{E}_{\mathscr{A},x,b}(\lambda)$:

$$\mathsf{E}_{\mathscr{A},x,b}(\lambda) : \mathscr{A}^{f_0(.), f_1(.), \ldots, f_n(.)}(1^\lambda, x, y_b) = b$$

where y_b satisfies the following:

- $\mathsf{dist}(y_b, f_0(x)) \leq \delta$ whenever $b = 0$
- uniformly random from $\mathscr{Y}$ whenever $b = 1$

and $\mathscr{A}$ is not allowed to issue a query of the form $f_0(x)$ and $f_i(x), 1 \leq i \leq n$ for $x \in \mathscr{X}_i$. The function f_0 is pseudo-random whenever any input $x \in \mathscr{X}$ satisfies the above conditions. Then, for any such PPT algorithm $\mathscr{A}$,

$$\left|\mathsf{Pr}[\mathsf{E}_{\mathscr{A},x,0}(\lambda)] - \mathsf{Pr}[\mathsf{E}_{\mathscr{A},x,1}(\lambda)]\right| \leq \mathsf{negl}(\lambda).$$

In the following definition, a ReF graph is defined as a collection of ReF where each ReF pair related over a unique set of inputs satisfies the property of reconciliation and pseudo-randomness.

ReF Graph. Assume $\mathscr{F} = \{f_i : \mathscr{X} \to \mathscr{Y}\}$ to be a family of functions for the arbitrary sets $\mathscr{X}$ and $\mathscr{Y}$. Then, a set of T functions $f_1, \ldots, f_\mathsf{T} \in \mathscr{F}$ represents a $(\mathsf{T}, \varepsilon, \varepsilon', \delta)-$ReF graph for $\varepsilon' \in (0, \varepsilon]$ whenever the following holds for each $i, j \in [\mathsf{T}]$:

- (f_i, f_j) is (ε, δ) related
- f_i is pseudorandom
- a unique subset of inputs ${\mathscr{X}'}_{i,j}$ exists with size at least $\varepsilon' |\mathscr{X}|$ for f_i and f_j.

Let **D** be a hardware device manufactured to physically realise the same input-output behaviour of the boolean function f. Then **D** is referred as reliable embodiment of function f.

Reliable Embodiment. **D** is a reliable physical embodiment of a function $f : \mathscr{X} \to \mathscr{Y}$ if :

$$\mathsf{Pr}[\mathbf{D}(x) \neq f(x)] \leq \mathsf{negl}(\lambda)$$

for all $x \in \mathscr{X}$ and the probability is defined on multiple calls to the device **D.**

Physically Related Function(PReF). A device pair $(\mathbf{D}_i, \mathbf{D}_j)$ represent an $(\varepsilon, \delta)-$physically related function , if it reliably embodies an (ε, δ) pair (f_i, f_j).

PReF Network. A set of devices $\mathbf{D}_1, \ldots, \mathbf{D}_\mathsf{T}$ represent a $(\mathsf{T}, \varepsilon, \varepsilon', \delta)-$PReF network whenever every device $\mathbf{D}_i$ reliably embodies a function f_i in a way that the corresponding collection of functions $f_1, \ldots, f_\mathsf{T} \in \mathscr{F}$ in turn represent a $(\mathsf{T}, \varepsilon, \varepsilon', \delta)-$ReF graph.

A.2 Hardware Realisation of PReFs.

Note that PReFs are fundamentally similar to PUFs except for the assumption on the existence of related input set to arrive at a common output. Thus their realisation in hardware also rely on the natural device specific variations to enable unpredictability on the input/output behaviour of the device even for an adversary possessing the complete knowledge of circuit implementation. Thus a natural constraint in the hardware realisation of PReFs is to identify related input set. As the definition A.1 allows the outputs of PReF pairs to be non identical but related to each other through a distance function, the related input subset can be obtained from any arbitrary random boolean functions. Chatterjee *et al.* proposed machine learning based methodology to obtain related inputs for a given hardware circuit realised with a pair of boolean functions in [9].

B PReF Based Key Exchange Protocol

Chatterjee *et al.* introduced a basic two party based key exchange protocol meant to establish secure communication by avoiding the use of traditional cryptographic assumptions.

System model for Static PReF Network. Let $\mathscr{F} = \{f_i : \mathscr{X} \to \mathscr{Y}\}$ be a family of functions, where $\mathscr{X} = \{0,1\}^m$ and $\mathscr{Y} = \{0,1\}^n$ are arbitrary sets. Let Alice and Bob be two parties that physically implement f_1 and f_2 using their respective hardware devices D_1 and D_2. Assume that there exist a trusted party (e.g., a manufacturer) which is semi-honest and does the following before the deployment of devices $D_i, i = 1, \ldots, T$ in the static PReF network with the corresponding functions $f_i, i = 1, \ldots, T$: (a) A $(\mathsf{T}, \varepsilon, \varepsilon', \delta)-$ReF graph represented by the functions $(f_1, f_2, \ldots, f_\mathsf{T}) \in \mathscr{F}$ is constructed where each f_i is embodied to the corresponding device $D_i, i = 1, \ldots, T$. (b) It also publishes the related input set $\mathscr{X}_{i,j}$ for the corresponding devices D_i and D_j on a public loud/ledger. The parties D_1 and D_2 communicate on an input x from a publicly available related input set $\mathscr{X}_{1,2}$ and compute the output y using their respective PReFs f_1 and f_2. Note that an adversary with f_0 cannot predict the value y even if it shares PReF pairs $\mathscr{X}_{1,0}$ and $\mathscr{X}_{2,0}$ as

$$\mathscr{X}_{1,2} \cap \mathscr{X}_{1,0} = \mathscr{X}_{1,2} \cap \mathscr{X}_{2,0} = \phi.$$

$\mathscr{D}_1$		$\mathscr{D}_2$
select a random $x \in \mathscr{X}_{1,2}$		
random $r \in \{0,1\}^n$		
Compute $y_1 = f_1(x)$		
$A = E(r) + y_1$	$\xrightarrow{x, A}$	Compute $y_2 = f_2(x)$
		$r' = D(A + y_2)$
session key $\mathsf{SK} = r = r'$.		

Figure 4. Basic Key Exchange from PReFs [9]

Chatterjee *et al.*'s Basic Key Exchange from PReFs. The basic key exchange protocol using PReF in Figure 4 is executed to establish a session key between two parties D_1 and D_2 where the party D_1 initiates the session as follows: D_1 selects x at random from the input set $\mathscr{X}_{1,2}$ and then computes y_1 from x using f_1. Then D_1 computes A using y_1 and random r and sends (A,x) to D_2. Upon receiving (A,x), D_2 first finds y_2 as $f_2(x)$ and then obtains r using A and y_2. Finally the session key is set as r. Note that the protocol makes use of only physically related function (existence of PUF) and linear Error correcting codes as per Definition A.1 to establish the session key between two parties in one round of communication. The protocol does not require secure storage contrary to the requirement of the scheme proposed in [32] that: (1) uses fuzzy extractors for correcting the noisy response and then match it with the response stored in a server and (2) is vulnerable to attacks as described in [12]. To avoid the possibility of such security vulnerabilities, the protocol in Figure 4 uses BCH-based error correction to reconcile the outputs of PReF from both the devices and then engage in fixation of session key. However the protocol in Figure 4 is secure only against passive adversaries.

Upgraded AKE. To achieve security against replay, impersonation and integrity attacks, the basic protocol in Figure 4 is proposed as depicted in Figure 5 by incorporating a collision-resistant hash function $\mathscr{H} : \{0,1\}^* \to \{0,1\}^n$. Note that it is computationally infeasible to sample $\mathsf{TS}' \neq \mathsf{TS}$ and $r \neq r'$ such that $\mathscr{H}(r||\mathsf{TS}||\mathscr{D}_1||\mathscr{D}_2) = \mathscr{H}(r'||\mathsf{TS}'||\mathscr{D}_1||\mathscr{D}_2)$. Thus with the minimal addition to the basic PReF protocol, the upgraded protocol achieves the security guarantees of [33–35].

$\mathscr{D}_1$		$\mathscr{D}_2$
nonce TS		
select a random $x \in \mathscr{X}_{1,2}$		
random $r \in \{0,1\}^n$		
Compute $y_1 = f_1(x)$		
$B = \mathscr{H}(r\|\|\mathsf{TS}\|\|\mathscr{D}_1\|\|\mathscr{D}_2)$		
$A = E(r) + y_1$	$\xrightarrow{x,A,B,\mathsf{TS}}$	Compute $y_2 = f_2(x)$
		$r' = D(A + y_2)$
		Check if $B = \mathscr{H}(r'\|\|\mathsf{TS}\|\|\mathscr{D}_1\|\|\mathscr{D}_2)$
	$\xleftarrow{C}$	computes $C = \mathscr{H}(r'\|\|\mathsf{TS}\|\|\mathscr{D}_2\|\|\mathscr{D}_1)$
Check if $C = \mathscr{H}(r\|\|\mathsf{TS}\|\|\mathscr{D}_2\|\|\mathscr{D}_1)$		
session key $\mathsf{SK} = r = r'$.		

Figure 5. Upgraded AKE protocol using PReFs [9]

Chatterjee *et al.* [9] pointed out that the upgraded protocol depicted in Figure 5 does not achieve forward secrecy as x values are sent in clear. The adversary having black box access to the PReF oracle f_o that provides the output $y = f_o(x)$ whenever x is given, will learn the session key of the particular session using A. Also, the protocol restricts usage of x more than once as the adversary learns sum of two session keys $r + s$ whenever the same x is used in two sessions with different random values r and s.

$\mathscr{D}_1$		$\mathscr{D}_2$
nonce TS		
select a random $x \in \mathscr{X}_{1,2}$		
random $r \in \{0,1\}^n$, $\alpha \in [1, q-1]$		
Compute $y_1 = f_1(x)$		
$A = E(r) + y_1, b = \mathscr{H}'(r\|\|\mathsf{TS})$		
$U = \alpha bP$, $B = \mathscr{H}(r\|\|\mathsf{TS}\|\|U\|\|\mathscr{D}_1\|\|\mathscr{D}_2)$	$\xrightarrow{x,A,U,B,\mathsf{TS}}$	Compute $y_2 = f_2(x)$
		$r' = D(A + y_2)$
		Check if $B = \mathscr{H}(r'\|\|\mathsf{TS}\|\|U\|\|\mathscr{D}_1\|\|\mathscr{D}_2)$
		$\beta \in [1, q-1], b = \mathscr{H}'(r'\|\|\mathsf{TS})$
		computes $V = \beta bP$, $C = \mathscr{H}(r'\|\|\mathsf{TS}\|\|\mathscr{D}_2\|\|\mathscr{D}_1)$
Check if $C = \mathscr{H}(r\|\|\mathsf{TS}\|\|V\|\|\mathscr{D}_2\|\|\mathscr{D}_1)$	$\xleftarrow{V,C}$	session key $K = \beta U$
session key $K = \alpha V$		

Figure 6. Forward Secure AKE protocol using PReFs [9]

Forward Secure AKE. In order to achieve forward secrecy, Chatterjee *et al.* [9] proposed to incorporate elliptic curve Diffie–Hellman based protocol to compute session keys using the ephemeral values α and β. The protocol is illustrated in Figure 6.

References

[1] Brian LaMacchia, Kristin Lauter, and Anton Mityagin. Stronger security of authenticated key exchange. In *Provable Security*, pages 1–16, 2007.

[2] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *Advances in Cryptology — EUROCRYPT 2001*, pages 453–474, 2001.

[3] Andrew J. Clark. Physical protection of cryptographic devices. In *Advances in Cryptology - EUROCRYPT '87, Workshop on the Theory and Application of of Cryptographic Techniques*, volume 304, pages 83–93, 1987.

[4] Muhammad Aman, Kee Chua, and Biplab Sikdar. Position paper: Physical unclonable functions for iot security. In *the 2nd ACM International Workshop*, pages 10–13, 2016.

[5] Ulrich Rührmair. Towards secret-free security. In *Cryptology ePrint Archive, Paper 2019/388*, 2019.

[6] Mohammad Nasim Imtiaz Khan and Swaroop Ghosh. Comprehensive study of security and privacy of emerging non-volatile memories. *Journal of Low Power Electronic Applications*, 11(4), 2021.

[7] Sergei Skorobogatov. Semi-invasive attacks-a new approach to hardware security analysis. In *University of Cambridge, UK*, 2005.

[8] Durba Chatterjee, Harishma Boyapally, Sikhar Patranabis, Urbi Chatterjee, Debdeep Mukhopadhyay, and Aritra Hazra. Physically related functions: A new paradigm for light-weight key-exchange . *Cryptology ePrint Archive*, 2021/389.

[9] Durba Chatterjee, Harishma Boyapallya, Sikhar Patranabis, Urbi Chatterjee, Aritra Hazra, and Debdeep Mukhopadhyay. Physically related functions: Exploiting related inputs of pufs for authenticated-key exchange. *IEEE Transactions on Information Forensics and Security*, 17:3847–3862, 2022.

[10] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *Advances in Cryptology — CRYPTO' 88, LNCS*, volume 403, pages 147–160, 1988.

[11] R. W. Hamming. Error detecting and error correcting codes. *The Bell System Technical Journal*, 29:147–160, 1950.

[12] Jeroen Delvaux, Roel Peeters, Dawu Gu, and Ingrid Verbauwhede. A survey on lightweight entity authentication with strong pufs. In *ACM Computer Survey*, pages 1–42, 2015.

[13] Ali Shahidinejad and Jemal Abawajy. An all-inclusive taxonomy and critical review of blockchain-assisted authentication and session key generation protocols for iot. *ACM Comput. Surv.*, 56(7), April 2024.

[14] Gildas Avoine. Symmetric-key authenticated key exchange (SAKE) with perfect forward secrecy. In *Topics in Cryptology - CT-RSA 2020 - The Cryptographers' Track, LNCS*, volume 12006, pages 199–224, 2020.

[15] Suman Bala, Gaurav Sharma, and Anil Verma. Pf-id-2paka: Pairing free identity-based two-party authenticated key agreement protocol for wireless sensor networks. In *Wireless Personal Communications*, volume 87, 2015.

[16] S. Blake-Wilson, D. Johnson, and A.Menezes. Key agreement protocols and their security analysis. In *Cryptography and Coding 1997, LNCS*, volume 1355, 1997.

[17] C.Saint Guilhem, N. P. Smart, and B. Warinschi. Generic forward-secure key agreement without signatures. In *Information Security (ISC) 2017, LNCS*, volume 10599, 2017.

[18] Mohammad Sadeq Dousti and Rasool Jalili. Forsakes: A forward-secure authenticated key exchange protocol based on symmetric key-evolving schemes. *Advances in Mathematics of Communications,2015*, 9(4):471–514, 2015.

[19] Quanrun Li, Ching-Fang Hsu, Kim-Kwang Raymond Choo, and Debiao He. A provably secure and lightweight identity-based two-party authenticated key agreement protocol for vehicular ad hoc networks. *Security and Communication Networks*, 7871067, 2019.

[20] Xiruo Liu, Meiyuan Zhao, Sugang Li, Feixiong Zhang, and W. Trappe. A security framework for the internet of things in the future internet architecture. *Futue Internet*, 9, 2017.

[21] Ni Liang, Chen Gongliang, Li Jianhua, and Yanyan Hao. Strongly secure identity-based authenticated key agreement protocols. *Computers and Electrical Engineering*, 37:205–217, 2011.

[22] Dmitry Obukhov and Bin Tan. Encryption key destruction for secure data erasure. *United States Patent (US 9,467.288 B2) to SEAGATE TECHNOLOGY LLC*.

[23] Jingwei Liu, Qian L, Rui Yan, and Rong Sun. Efficient authenticated key exchange protocols for wireless body area networks. *J Wireless Com Network*, 188, 2015.

[24] Muhammad Naveed Aman and Kee Chaing Chuaand Biplab Sikdar. Mutual authentication in iot systems using physical unclonable functions. *IEEE Internet of Things Journal*, 4(5):1327–1340, 2017.

[25] Urbi Chatterjee, Rajat Subhra Chakraborty, and Debdeep Mukhopadhyay. A puf-based secure communication protocol for iot. *ACM Trans. Embed. Comput. Syst.*, 16(3), April 2017.

[26] Urbi Chatterjee, Vidya Govindan, Rajat Sadhukhan, Debdeep Mukhopadhyay, Rajat Subhra Chakrabortyand Debashis Mahata, and Mukesh M. Prabhu. Building puf based authentication and key exchange protocol for iot without explicit crps in verifier database. *IEEE Transactions on Dependable and Secure Computing*, 16(3):424–437, 2019.

[27] Jin Wook Byun. End-to-end authenticated key exchange based on different physical unclonable functions. *IEEE Access*, 7:102951–102965, 2019.

[28] Sensen Li, Tikui Zhang, Bin Yu, and Kuan He. A provably secure and practical puf-based end-to-end mutual authentication and key exchange protocol for iot. *IEEE Sensors Journal*, 21(4):5487–5501, 2021.

[29] Yue Zheng and Chip-Hong Zheng. Secure mutual authentication and key-exchange protocol between puf-embedded iot endpoints. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5, 2021.

[30] Harishma Boyapally, Durba Chatterjee, Kuheli Pratihar, Sayandeep Saha, Debdeep Mukhopadhyay, and Shivam Bhasin. Harmonizing PUFs for forward secure authenticated key exchange with symmetric primitives. Cryptology ePrint Archive, Paper 2024/484, 2024.

[31] Jeroen Delvaux, Roel Delvaux, Dawu Gu, and Ingrid Verbauwhede. A survey on lightweight entity authentication with strong pufs. *ACM Comput. Surv.*, 48(2), October 2015.

[32] Roel Maes, Roel Peeters, Anthony Herrewege, Christian Wachsmann, Stefan Katzenbeisser, Ahmad-Reza Sadeghi, and Ingrid Verbauwhede. Reverse fuzzy extractors: Enabling lightweight mutual authentication for puf-enabled rfids. In *Financial Cryptography*, pages 374–389, 2012.

[33] Ran Canetti, Daniel Shahaf, and Margarita Vald. Universally composable authentication and key-exchange with gluobal pki. In CM Cheng, K M Chung, G Persiano, and B Y Yang, editors, *Public-Key Cryptography – PKC 2016, LNCS.*, volume 9615, 2016.

[34] L. Law, A. Menezes, M. Qu, J. Solinas, and Vanstone S. Error detecting and error correcting codes. *Designs Codes and Cryptography*, 28:119–134, 2003.

[35] Bin Rabiah, Abdulrahman, K Ramakrishnan, Olule-Liri, Elizabeth, and Koushik Kar. A lightweight authentication and key exchange protocol for iot. In *Workshop Decentralised IoT Security Standards*, 2018.