TITLE: Response-Based Cryptography with Physical Unclonable Functions

INVENTORS: Bertrand F. Cambou, Christopher Philabaum, Dennis D. Booher

CROSS REFERENCE TO RELATED APPLICATIONS AND PUBLICATIONS

This is the original provisional patent application. All references mentioned in this application are herein incorporated by reference without disclaimer.

FIELD OF THE INVENTION

The invention relates to cybersecurity and authentication systems. More particularly, the invention relates to applications utilizing physically unclonable functions (PUFs) in cryptographic protocols.

BACKGROUND OF THE INVENTION

Conventionally, physically unclonable functions (PUFs) form a set of cryptographic primitives used in authentication methods. The underlying authentication mechanism in which PUFs are used is in generating challenge-response pairs (CRPs). PUFs that are unique to each device allow an authentication system to challenge a device seeking authentication, receive half of a CRP pair from the device, and then compare the received half with a stored CRP pair to verify whether a match takes place. Where the PUFs take advantage of the natural manufacturing variations unique to the device seeking authentication, a large number of challenge (i.e., input)-response (i.e., output) pairs can be generated. The generation of CRPs may need to be reproducible, predictable, and easy to recognize during the authentication process for the CRPs to be useful.

SUMMARY OF THE INVENTION

[ADD SUMMARY OF THE CLAIMS HERE]

BRIEF DESCRIPTION OF THE DRAWINGS

The drawings described herein constitute part of this specification and includes exemplary embodiments of the present invention which may be embodied in various forms. It is to be understood that in some instances, various aspects of the invention may be shown exaggerated or enlarged to facilitate an understanding of the invention. Therefore, drawings may not be to scale.

FIG. 1 depicts the enrollment wherein the PUF challenges of the client devices are downloaded into a look-up table, according to one emodiment.

FIG. 2 presents a block diagram of a network protected by PUF-based cryptography, according to one embodiment.

FIG. 3 depicts a block diagram of a network secured by response-based cryptography (RBC), according to one embodiment.

FIG. 4 shows an authentication of client devices, according to one embodiment.

FIG. 5 shows authentication at different levels of Hamming distances, according to one embodiment.

FIG. 6 depicts a response-based key matching algorithm, according to one embodiment.

FIG. 7 depicts a modeling of the time needed to encrypt a group of 256-bit streams with AES and compare with a reference cipher wherein the Hamming distance varies from 0 to 5.

FIG. 8 is a table of trade-offs to minimize RBC latencies for various CRP error rates.

FIG. 9 depicts a testing of RBC with a commercial SRAM, according to one embodiment.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Authentication protocols based on various physical unclonable functions (PUFs), embedded in each Internet of Things (IoT) node, can be effective when the drifts in the PUF characteristics are small enough. This invention employs the following definitions:

The term “client devices” refers to the components secured by the PUFs wherein said devices may be IoT devices or other peripheral devices.

The term “server” is the component driving the set of client devices.
The term “challenges” are the initial data streams generated during enrollment of the client devices by the server. In some protocols, the challenges can be generated by processing or averaging multiple queries and measurements of the PUFs. The challenges can also be the result of computations and/or statistical analysis. The challenges can be represented by binary, ternary, or other arithmetic bases.

The term “enrollment” of the PUFs is the operation when the challenges of the PUFs located in each client device are downloaded in a database or look-up table in the server. The operation is generally done in a secure environment. A block diagram is presented in FIG. 1. Additionally, new PUFs can be enrolled over time.

The term “responses” are the data streams generated by the PUFs during the life of the client devices. These PUFs are physical elements that can age and can be subject to temperature changes, electromagnetic interferences, and other environmental effects.

The term “challenge-response pairs” (CRPs) are generated on demand by the server of the PUFs of the client devices.

When a PUF is a strong PUF with multiple PUFs, the server needs to send “instructions” to the client devices to find the particular “address” in the PUF and to generate challenge-response pairs.

The term “helper” is the data stream generated by the server and communicated to the client devices to be able to correct the challenge-response pair error rates.

One area of confusion relates to the use of similar terms to describe PUFs. In some cases, the “instructions” are meant as challenges and the “challenges” are meant to describe the initial response.

In such cases, the server will send challenges which are used by the client to generate responses. The server will keep track of the initial responses and match it with the freshly generated responses. The error rates are therefore tracked between initial responses and freshly generated responses. The concept of CRP error rates can then be replaced by response-response pair error rates (i.e., initial response-fresh response pairs). Both terminologies are equivalent, and this invention is agnostic on which terminology is preferred.

When the CRP error rates are relatively low, the responses can be used as part of authentication protocols, which has significant commercial value to protect cyber physical systems. In such cases, Hamming distances between challenges and responses are in the 10%, range, and both the false acceptance rate (FAR) and false reject rates (FRR) can be low enough for satisfactory authentication. When the CRP error rates are too high, the use of error-correcting methods and helper improve both FAR and FRR.

The use of PUFs to generate cryptographic keys from the responses is more challenging than generating responses for authentication. This requires schemes that fully correct the responses of the PUF; a single-bit mismatch in the cryptographic keys is not acceptable for most encryption protocols. Ciphers, in particular block ciphers, cannot be decrypted with keys having a single-bit mismatch.

A typical architecture to drive a constellation of client devices with PUFs is shown in FIG. 2, according to one embodiment. As part of the handshake process, the server initiates the process by sending the instructions to the client device j that incorporate the address Address j1 within the PUF on where to extract responses Responses-j1. The server independently analyzes the challenges Challenges j1 stored in its look-up tables at the corresponding address and generates the helper, Helper j1, with fuzzy extractor and other error-correcting methods from Challenges j1. These helpers are transmitted to the client devices as part of the handshake. The client devices generate Responses j1 at the address, Address j1, and correct them with the helper with fuzzy extractors and other correcting methods. To be acceptable for encryption, the corrected responses of the client devices and the challenges of the server should be identical. Thereby, the same key, Key-j1, is independently generated for encryption schemes.

Such protocols have two fundamental weaknesses: first, the client devices are burdened and need to consume additional computing power to run the error-correcting codes; second, such protocols increase the vulnerability to side-channel attacks, differential power analysis, and potential exposure of the helpers.

The novelty of the methods presented herein is that there is no need to correct the PUF responses as the keys are generated directly from the uncorrected responses. The burden is transferred to the server, which can have access to considerable computing power and the highest level of security, while the clients can operate at low power in a non-secure environment.

Response-Based Cryptography (RBC)

The protocol described above and shown in FIG. 1 can be referred to as “challenge-based cryptography”; this invention describes a “response-based cryptography” (RBC) using uncorrected responses from a constellation of PUFs securing client devices. A block diagram of a network protected by a RBC protocol is shown in FIG. 2 according to one embodiment.

As done in the challenge-based cryptography, the secure server sends instructions to the client device j and the Addresses j1 to generate the PUF responses, Responses j1. The cryptographic key, Key-j1, is directly extracted from the responses.

The purpose of the RBC Engine, which is driven by the server, is to generate from the challenges, Challenges j1, and the responses matching the ones generated by the PUFs and to extract the exact same cryptographic key, Key-j1.

The server does not need to generate and transmit helper messages anymore, and the client devices do not need to have error-correcting schemes to extract data streams matching the challenges stored by the server. The computing power needed at the client level is thereby reduced, which allows the use of less powerful microcontrollers, smaller memory components, and simpler architectures.

The elimination of the helpers also simplifies the communication between server and clients. The latency at the client device level is significantly reduced, giving less time for malicious observers to extract relevant information from the transaction.

Authentication Protocol with RBC

During authentication, the client device encrypts a message for authentication purpose, MA-j1, with a scheme such as Advanced Encryption Standard (AES) or the like, with the cryptographic key, Key j1. The encrypted message, E(MA-j1, Key-j1), is transmitted to the server and analyzed by the RBC engine (FIG. 4). Examples of message for authentication purposes could be the user ID of client j or other identification scheme. The cipher, E(MA-j1, Key-j1), needs the cryptographic Key-j1 to be decrypted, which cannot be done by malicious parties attacking the network without this key. For most protocols, similar authentication schemes are needed to prevent unauthorized client to interact with the network and may also be used for response-based cryptography.

Responses j1-0 are defined as the data stream having a Hamming distance of zero with Challenge j1, which means Challenge j1 = Response j1. The key, Kj1-0, is extracted from Response j1-0 and is used to encrypt MA-j1 to generate the cipher E(MA-j1, Kj1-0) (FIG. 6). When the challenges, Challenges j1, and responses, Responses j1, have CRP error rates at zero, and the authentication of the client device j is positive:

$$E(MA\text{-}j1, Key\text{-}j1) = E(MA\text{-}j1, Kjl\text{-}0) \qquad \text{[Equation 1]}$$

$$Key\text{-}j1 = Kj\text{-}0 \qquad \text{[Equation 2]}$$

If the CRP error rate is not zero, Equation 1 is false. Responses j1-1-k are defined with $k \in \{1, x\}$, all data streams with Hamming distances with Challenges j1 of one. For example, if the data streams are 256-bits long, there are 256 streams having Hamming distance of 1 with Challenge j1 and x=256. The keys, Kj1-1-k with $k \in \{1, x\}$, are generated from these streams, are used to encrypt MA-j1, and then generate the ciphers E(MA-j1, Kj1-1-k), with $k \in \{1, x\}$. If the CRP error rate between Challenge j1 and Response j1 is one, then one of these ciphers will be equal to E(MA-j1, Key-j1). The response Responses j1-1-k, which generates the cipher equal to E(MA-j1, Key-j1) is therefore equal to Response j1, and the authentication of j is positive:

$$Key\text{-}j1 = Kj1\text{-}1\text{-}k \qquad \text{[Equation 3]}$$

If the CRP error rate between Challenge j1 and Response j1 is two, one of the ciphers, E(MA-j1, Kj1-2-k) with k ϵ {1, y}, is equal to E(MA-j1, Key-j1). The response, Responses j1-2-k, which generates the cipher equal to E(MA-j1, Key-j1) is therefore equal to Response j1, and the authentication of j is positive:

Key-j1 = Kj1-2-k [Equation 4]

In a similar way, if the CRP error rate between Challenge j1 and Response j1 is three, one of the ciphers, E(MA-j1, Kjl-3-k) with k ϵ {1, z}, is equal to E(MA-j1, Key-j1}. The response Responses j1-3-k, which generates the cipher equal to E(MA-j1, Key-j1) is therefore equal to Response j1, and the authentication of j is positive:

Key-j1 = Kj1-3-k [Equation 5]

The process described above, which is summarized in FIG. 5, can be extended to the generation of the data streams having Hamming distances higher than 3. The computing power needed for such authentication process is described further on.

Basic Protocol of Response-Based Cryptography

Increased Hamming Distance. The authentication protocol described above is the first step of the protocol driven by the RBC engine. Assuming that the computing power of the server device is infinite, the authentication process can iterate to find a data stream having the same cipher as the client, E(MA-j1, Key-j1) (FIG. 6). The trade-off computing power at various levels of PUF quality is described below. When their ciphers are matching, both parties share the same cryptographic key Key-j1, which can be used to protect the communication between client-server. In such a protocol, which is based on increasing the Hamming distance systematically, the entire burden is placed on the server. However, infinite computing power does not exist; this protocol is limited to PUFs with CRP error rates that are low enough and have small Hamming distances between challenges and responses.

Multiple queries. The frequency of PUF CRP errors is highly variable. A bad connection server to client, a noisy environment, an abrupt temperature variation, or simply the selection of a marginal section of the PUF can result in high CRP rates. Thereby, rather than trying to find a match with high Hamming distances, it is faster at some point to stop the process described above and initiate a new address, Address j2.

This revised protocol places part of the burden on the client devices; a new set of responses, keys, and encrypted messages are then generated. The CRP error rate will vary with each query; multiple queries will reduce the time needed to find matching keys. For example, with a 256-bit long PUF having an average CRP error rate of 0.03% and limited computing power, 10 seconds is necessary to find a match with a single query, as opposed to only 2ms are needed with two queries.

Error correction. The time and computing power needed to reduce the errors of a response is relatively small if the expected error rate post correction is not zero. For example, the use of ternary cryptography on PUFs is effective in reducing CRP error rates below 0.03% without burdening client devices. The combination of "light" correcting methods with "increased Hamming distance" protocols, and "multiple queries" make RBC effective.

Modeling and Variation

Hamming Distance of RBC. In order to demonstrate the practicality of RBC, it must be estimated how much time is needed to generate all response streams with a distance, a, and to encrypt them for the purpose of matching the ciphers with E(MA-j1, Key-j1). If we assume that the length of the challenges is 256-bit, the number of possible streams, Ns, with Hamming distance of "a" is given by:

$$Ns = \binom{256}{a} \qquad \text{[Equation 6]}$$

For each stream "k", the cipher E(MA-j1, Kj1-a-k) is generated with encryption schemes such as AES and compared with E(MA-j1, Key-j1). FIG. 7 shows the result of this modelling. The time needed to process all streams when "a" increases from a=O to a= 5. At a given Hamming distance, this time is further reduced when more powerful servers are used. This simple model shows that the time needed exponentially increases with a. On average, the time needed at the last round will halve, and there are methods to reduce the time by applying parallel methods, and entanglements.

Efficiency of RBC at Various Error Rates. Repeating a query is faster than searching a match with Hamming distance greater than 1; FIG. 8 shows the analysis of the efficiency of RBC at different levels of CRP error rates.

Failure rates with Several Hamming Distances. As shown on the left side of the table of FIG. 8, the efficiency of RBC, using the Hamming distance method described above, is poor when the CRP error rates are 1%, or higher, and is strong when the CRP error rates are below 0.1%.

Combination with Queries. The question analyzed in the mid-section of FIG. 8 is the respective efficiency of multiple queries versus increasing "a". Purely from a security standpoint, it is better to reduce the interactions client-server (i.e., reduce the number of queries); however the server needs to be able to quickly find a match. For example, if the CRP error rate is 1%, seven queries are needed for the RBC with a=2, assuming a FRR lower than 0.1% to authenticate a PUF; six queries are needed with a=3, three queries with a=4, and only two queries with a=5.

The section on the right of FIG. 8 is a modeling of the latency of RBC with AES scheme. The latency is always lower when the application allows multiple queries. In the example of CRP error rate of 1% and FRR < 0.1%, it takes a server 7ms to authenticate a client device with 7 queries and a=2, and as much as two minutes with 2 queries and a=S. However, when the CRP error rate is lower, for example 0.03%, a server can authenticate a client device in only 1 query at 100ms with a=3.

Experimental Evaluation

A 32kByte commercial SRAM produced by Cypress was analyzed (FIG. 9). SRAM-based PUFs exploits the fact that their cells are designed with flip-flop logic. During power-off/power-on cycles, a large proportion of the cells flip as a "0" or as a "1" due to manufacturing variations of the nanomaterials. Other cells are more symmetrical and randomly flip on both sides. Each SRAM device is different from others; the patterns of the cells are PUF challenges/responses.

The CRP error rates of these PUFs are reduced when the cells that are random are eliminated. As shown in FIG. 9, the CRP error rate is reduced when the random cells are eliminated, and the SRAM is tested multiple times. At each event, the random cells are removed, and the error rates of the remaining cells are evaluated. After only three cycles, the CRP error rate of the remaining cells is dropping below 1%, after 8 cycles the CRP drop below 0.3%, after 27 cycles this drop to 0.1%, and to 0.03% after 47 cycles.

In this protocol with the use of ternary states, at least 50 cycles are performed to have a PUF with CRP error rates below 0.03%. At that level, RBC authentication with 0.1% FRR can be done within 1μs (2 queries, a=1), or 100ms (1 query, a=3), which is extremely fast. This experimental work fully demonstrates the practicality of RBC protocol.

The described features, advantages, and characteristics may be combined in any suitable manner in one or more embodiments. One skilled in the relevant art will recognize that the circuit may be practiced without one or more of the specific features or advantages of a particular embodiment. In other instances, additional features and advantages may be recognized in certain embodiments that may not be present in all embodiments.

Reference throughout this specification to “one embodiment,” “an embodiment,” or similar language means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment. Thus appearances of the phrase “in one embodiment,” “in an embodiment,” and similar language throughout this specification may, but do not necessarily, all refer to the same embodiment.

CLAIMS

The invention claimed is:

1. A cryptographic method controlled by a first device, the "Server", to authenticate a second device, the "Client", to encrypt and decrypt messages between the two devices. The Client contains physical unclonable functions (PUF), having their initial readings, the "Challenges" stored in the Server. The method follows the following steps:

> 1.1 During authentication, or encryption cycles the Server send instructions to the Client to address their PUF, and generate new readings, the "Responses". The authentication of the Client, and the generation of cryptographic keys for encryption are generated directly from the Responses. The Client send an identification message to the Server containing a stream that is known to the Server, which is encrypted with a cryptographic key extracted from the PUF Responses.
>
> 1.2 The Server generate a set of multiple streams from the original challenges, encrypt them with the known identification message, and compare the ciphers with the one transmitted by the Client. The Challenges are the original readings of the PUF that can be found in the database of the Server following independently the same instructions than the ones described in claim 1.1. If the matching between both ciphers is negative, the Server start an iterative process, and try to find a matching cipher with all streams with a Hamming distance of one with the Challenges. If the matching is still negative the Server try the streams with higher Hamming distance to find the stream that generates the same cipher.
>
> 1.3 The matching stream extracted by the server, as described in claim 1.2, is used to authenticate the Client, and generate the cryptographic keys that allows secure encrypted communication between Server, and Client. The Client continues to use the Responses of claim 1.1 for its side of the encryption protocols;

2. Wherein the number of iterative attempts described claim 1.2 is limited to a Hamming distance of limited size. In case of a failure to find a matching cipher, the Server send a different instruction to the Client, generate different Responses, and repeat the same protocol as long as necessary to find a matching cipher;

3. Wherein ternary cryptography, error detecting, error correcting methods, fuzzy extractors, machine learning, artificial intelligence, are used in combination of the methods described in claims 1-2;

4. Wherein the encryption methods described claims 1-3 are, not to be limited to, AES, DES, Blowfish, RSA, Elliptic Curve Cryptography, Diffie-Helman Cryptography, One-time pads, Digital Signatures Algorithms, blockchains, Hash functions, and the combination off;

5. Wherein the physical unclonable functions of claim 1 are based on, not limited to, SRAM, DRAM, NOR Flash, NANO Flash, Ring oscillators, gate delays, Arbiters, Magnetic RAM, STIMRAM, Ferroelectric RAM, Phase Change RAM, Resistive RAM, memristors, OTP, EEPROM, and optical PUFs.

6. Wherein multi-factor authentication, and hash functions are used to protect the method described claims 1-5. The multi-factor authentication methods are, not to be limited with, password managers, pin codes, biometric prints, fingerprints, retina and iris prints.

ABSTRACT

Physical Unclonable functions (PUF) that are generated by the components of client devices, which can be used as fingerprints to authenticate these client devices, by using them to extract cryptographic keys which requires a powerful error correcting scheme. This invention describes methods to extract cryptographic keys directly from the un-corrected responses of the PUFs. The secure server driving the network manages the differences between the PUF responses, and the original PUF challenges, in such a way that the client devices do not need error correction scheme. As a result of the protocol both the server, and the client devices independently generate the same un-corrected responses of the PUF, which can be used in cryptographic protocols.

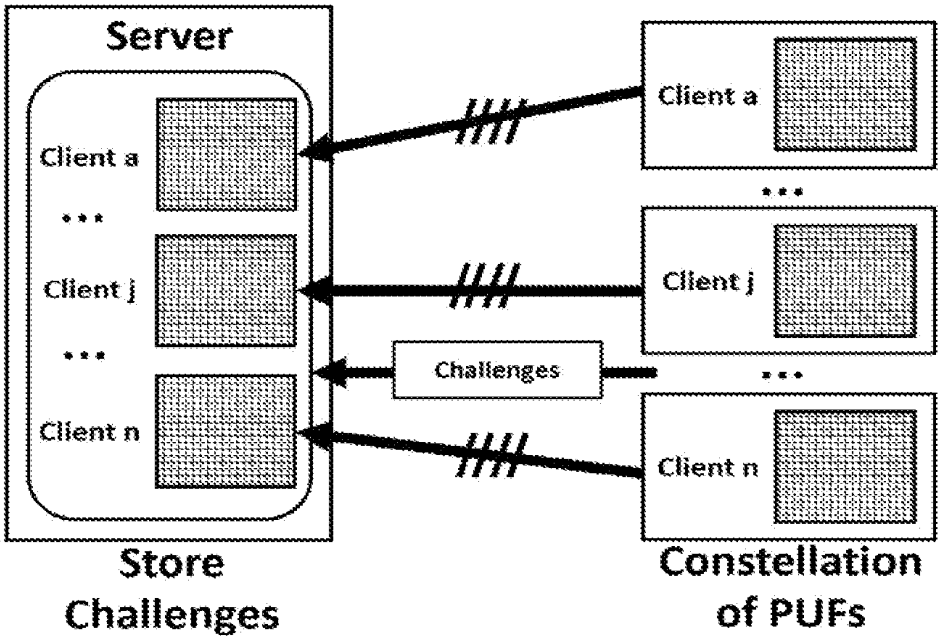
Server
Client a
Client j
Client n
Store Challenges
Challenges
Client a
Client j
Client n
Constellation of PUFs

FIG. 1

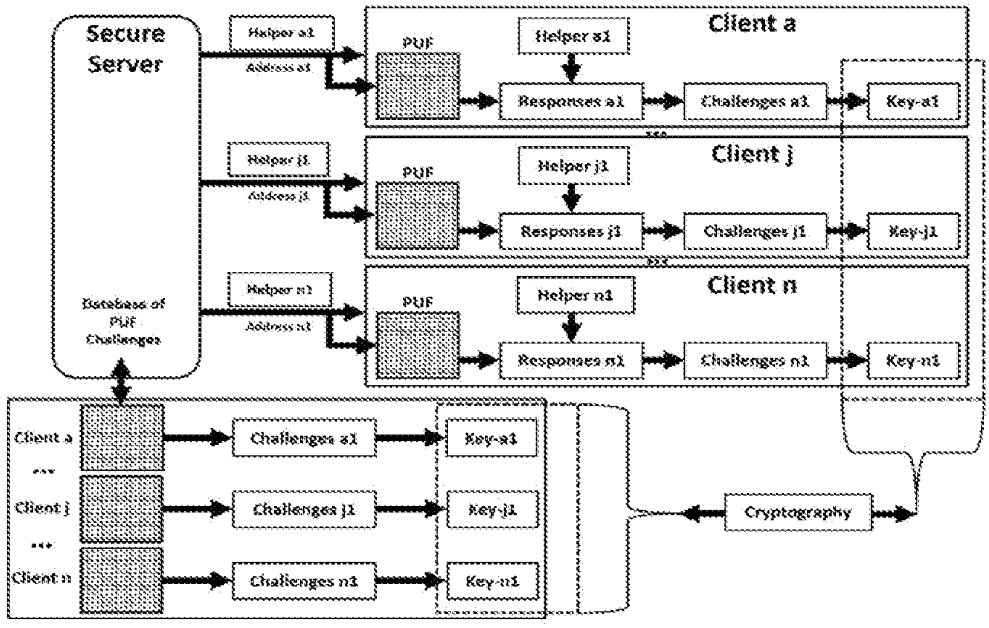
Secure Server
Database of PUF Challenges
Helper a1
Address a1
Helper j1
Address j1
Helper n1
Address n1
Client a
PUF
Helper a1
Responses a1
Challenges a1
Key-a1
Client j
PUF
Helper j1
Responses j1
Challenges j1
Key-j1
Client n
PUF
Helper n1
Responses n1
Challenges n1
Key-n1
Client a
Client j
Client n
Challenges a1
Challenges j1
Challenges n1
Key-a1
Key-j1
Key-n1
Cryptography

FIG. 2

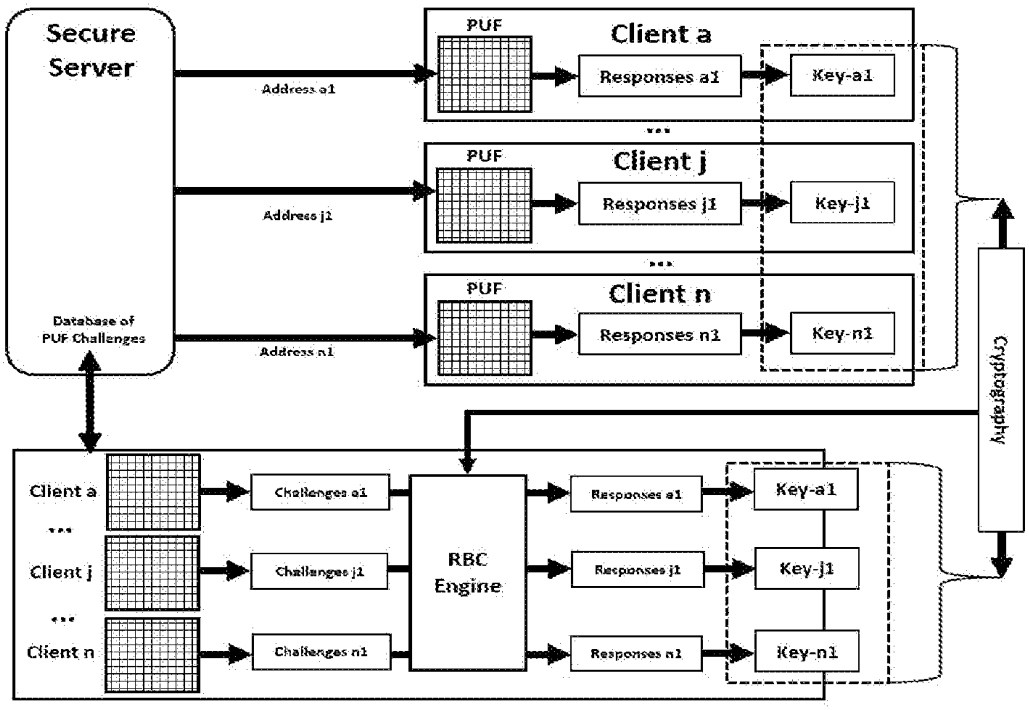
Secure Server
Database of PUF Challenges
Address a1
Address j1
Address n1
PUF
Client a
Responses a1
Key-a1
PUF
Client j
Responses j1
Key-j1
PUF
Client n
Responses n1
Key-n1
Cryptography
Client a
Client j
Client n
Challenges a1
Challenges j1
Challenges n1
RBC Engine
Responses a1
Responses j1
Responses n1
Key-a1
Key-j1
Key-n1

FIG. 3

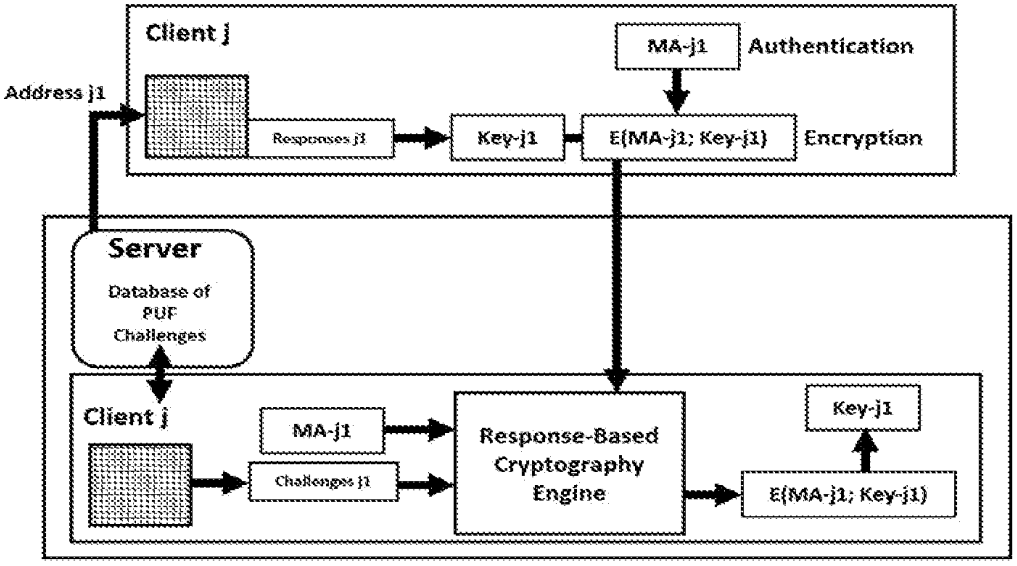
Client j
MA-j1
Authentication
Address j1
Responses j1
Key-j1
E(MA-j1; Key-j1)
Encryption
Server
Database of PUF Challenges
Client j
MA-j1
Challenges j1
Response-Based Cryptography Engine
Key-j1
E(MA-j1; Key-j1)

FIG. 4

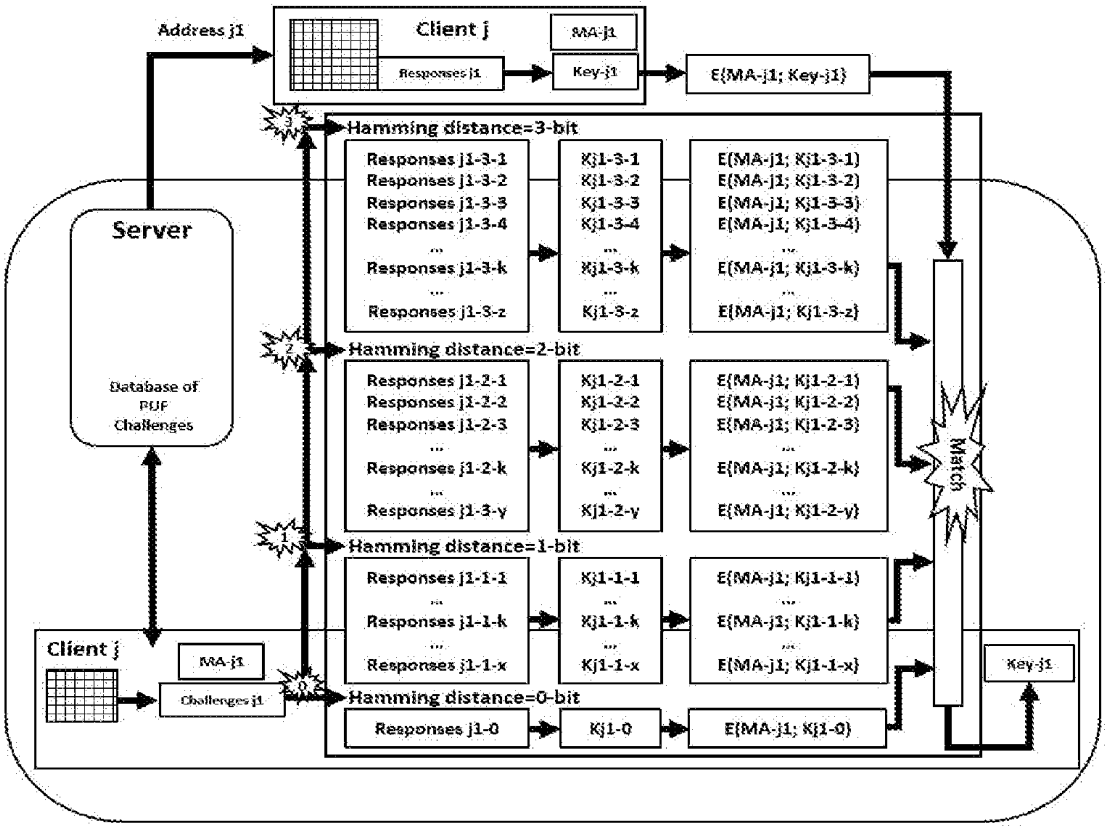
Address j1
Client j
MA-j1
Responses j1
Key-j1
E{MA-j1; Key-j1}
Hamming distance=3-bit
Responses j1-3-1
Responses j1-3-2
Responses j1-3-3
Responses j1-3-4
Responses j1-3-k
Responses j1-3-z
Kj1-3-1
Kj1-3-2
Kj1-3-3
Kj1-3-4
Kj1-3-k
Kj1-3-z
E{MA-j1; Kj1-3-1)
E{MA-j1; Kj1-3-2)
E{MA-j1; Kj1-3-3)
E{MA-j1; Kj1-3-4)
E{MA-j1; Kj1-3-k)
E{MA-j1; Kj1-3-z}
Server
Database of PUF Challenges
Hamming distance=2-bit
Responses j1-2-1
Responses j1-2-2
Responses j1-2-3
Responses j1-2-k
Responses j1-3-y
Kj1-2-1
Kj1-2-2
Kj1-2-3
Kj1-2-k
Kj1-2-y
E{MA-j1; Kj1-2-1)
E{MA-j1; Kj1-2-2)
E{MA-j1; Kj1-2-3)
E{MA-j1; Kj1-2-k)
E{MA-j1; Kj1-2-y)
Match
Hamming distance=1-bit
Responses j1-1-1
Responses j1-1-k
Responses j1-1-x
Kj1-1-1
Kj1-1-k
Kj1-1-x
E{MA-j1; Kj1-1-1)
E{MA-j1; Kj1-1-k}
E{MA-j1; Kj1-1-x}
Client j
MA-j1
Challenges j1
Hamming distance=0-bit
Responses j1-0
Kj1-0
E{MA-j1; Kj1-0)
Key-j1

FIG. 5

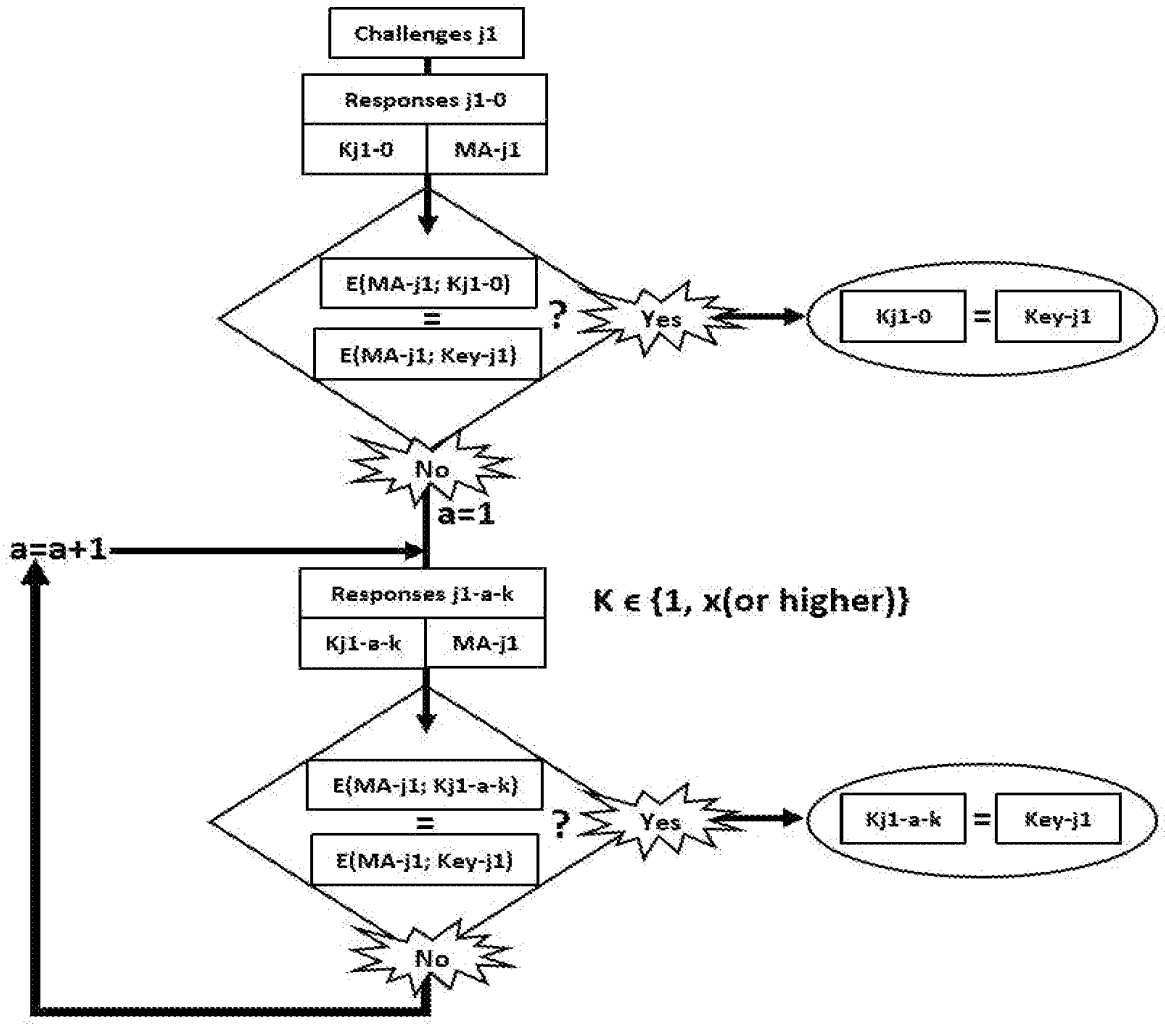
Challenges j1
Responses j1-0
Kj1-0
MA-j1
E(MA-j1; Kj1-0)
=
E(MA-j1; Key-j1)
?
Yes
Kj1-0
=
Key-j1
No
a=1
a=a+1
Responses j1-a-k
Kj1-a-k
MA-j1
K ∈ {1, x(or higher)}
E(MA-j1; Kj1-a-k)
=
E(MA-j1; Key-j1)
?
Yes
Kj1-a-k
=
Key-j1
No

FIG. 6

Hamming Distance "a"	Numbers of 256-bit Streams	Time: at 1GHz	Time: 32 Cores at 4GHz
0	$\binom{256}{0}$=1	5µs	< 1µs
1	$\binom{256}{1}$=256	1ms	1µs
2	$\binom{256}{2}$=32,512	100ms	1ms
3	$\binom{256}{3}$=2,763,520	10s	100ms
4	$\binom{256}{4}$=174,792,640	10min	1s
5	$\binom{256}{5}$=8,809,549,056	10hrs	1min

FIG. 7

256-bit Error Rate %	Failure Rate (FRR) in % to Match Responses with Several Hamming Distances "a"				Queries Needed for FRR < 0.1% (Only 0-bit Mismatch Accepted)					Latency for 1 or 2 Queries or More FRR < 0.1%					
	a=2	a=3	a=4	a=5	a=1	a=2	a=3	a=4	a=5	PC			Server		
										1Q	2Q	>	1Q	2Q	>
3	96	90	81	69	-	-	-	-	18	-	-	180hr	-		18min
1	36	19	8.4	3.3	-	7	6	3	2	-	20hr	700ms	-	2min	7ms
0.3	7.9	1.9	0.4	0.09	6	3	2	2	1	10hr	20s	6ms	1min	200ms	6µs
0.1	1.7	0.6	0.12	0.03	3	2	2	2	1	10hr	200ms	3ms	1min	1ms	3µs
0.03	0.15	0.01	$1x10^{-3}$	$1x10^{-4}$	2	2	1	1	1	10s	2ms		100ms	2µs	
0.01	$3x10^{-3}$	$1x10^{-4}$	$3x10^{-6}$	$1x10^{-7}$	1	1	1	1	1	1ms			1µs		

FIG. 8

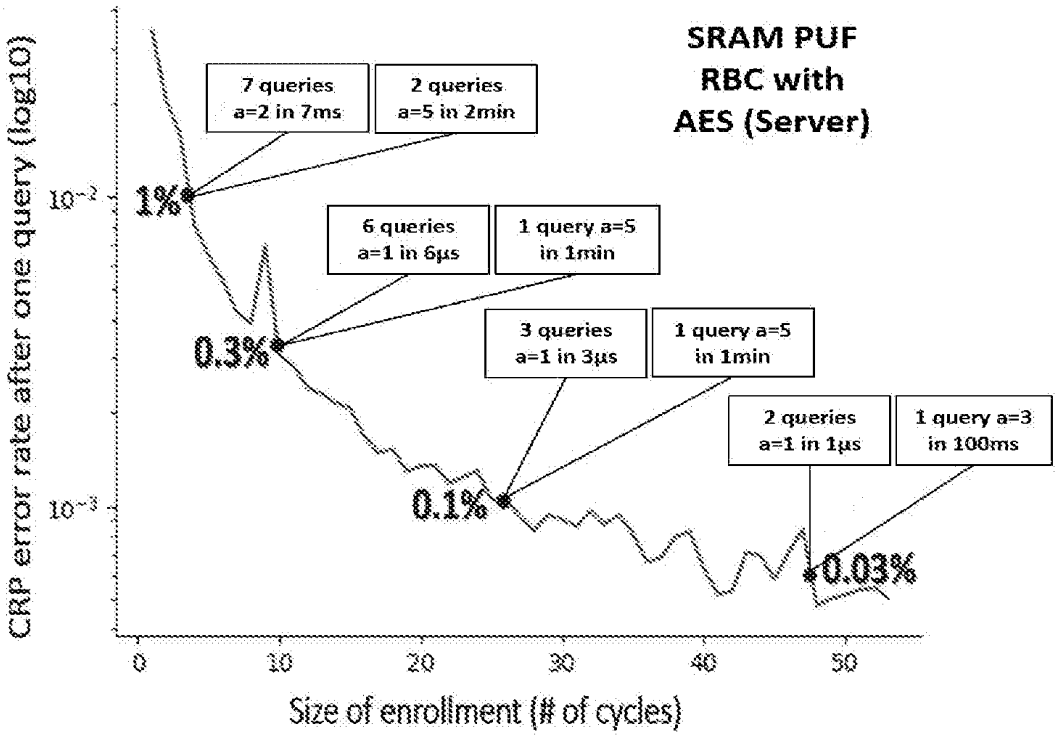
SRAM PUF
RBC with
AES (Server)
7 queries
a=2 in 7ms
2 queries
a=5 in 2min
1%
6 queries
a=1 in 6µs
1 query a=5
in 1min
0.3%
3 queries
a=1 in 3µs
1 query a=5
in 1min
2 queries
a=1 in 1µs
1 query a=3
in 100ms
0.1%
0.03%
CRP error rate after one query (log10)
10^{-2}
10^{-3}
0
10
20
30
40
50
Size of enrollment (# of cycles)

FIG. 9