TITLE: Digital Signature for Blockchains with Ternary Physically Unclonable Functions (PUFs)

INVENTORS: Bertrand Cambou

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 present invention generally relates to implementations of physically unclonable functions (PUFs) for cryptographic and authentication applications. More particularly, the invention relates to implementations of PUFs and blockchains.

BACKGROUND OF THE INVENTION

The blockchain technology with a hashing function, such as SHA-2, and digital signature, such as elliptic curve cryptography (ECC), has the potential to secure Internet of Things (IoT) infrastructure and protect the data flow needed to track transactions for applications such as strategic manufacturing, finance, and commerce. The underlying assumption is that the entire infrastructure of IoT is homogeneous with each node protected by a crypto-processor handling the hashing and having secure non-volatile memory to store the cryptographic keys. Malicious side-channel attacks as well as physical hijacking of the IoT nodes can expose the private keys, thereby compromising the security of the infrastructure. The secure distribution of public/private key pairs in such an environment can be risky. Without reliable protection of the private keys, the digital signature of the blockchains is vulnerable, and the technology loses its value.

Solutions based on physical unclonable functions (PUFs), embedded in each IoT node are attractive. Public/private key pairs are then generated on demand from the PUFs to secure the blockchains with a digital signature (DSA), thereby mitigating the risk that a malicious attacker compromise portions of the IoT infrastructure. The DSA can be based, not limited to, on finite field ECC (also called Galois Fields ECC: GF-ECC), which are lower power than regular ECC and RSA. However, the error correction schemes needed to handle the PUFs trend to add prohibitive complexity. With PUF challenge-response pairs (CRPs), a single bit mismatch that is not corrected creates a catastrophic mismatch of the public/private key pairs.

SUMMARY OF THE INVENTION

The present invention provides methods to use ternary PUFs to generate reliable public/private key pairs for the DSA of blockchains without using complex error correction. The methods described herein are a continuation in part of some of the prior art on ternary PUFs and associated cryptography. The main objective of the novel schemes is to generate error-free key pairs for asymmetrical cryptography from ternary PUFs and memory based PUFs.

[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 step wherein various PUF challenges are downloaded from the client devices to a look-up table securely stored by the server, according to one embodiment.

FIG. 2 is a block diagram of a client device with APG, according to one embodiment.

FIG. 3 depicts an overall architecture of the secure network, according to one embodiment.

FIG. 4 shows the CRP error rate of an SRAM array subjected to repetitive enrollment, according to one embodiment.

FIG. 5 depict the enrollment step using ternary PUFs with multiple readings, according to one embodiment.

FIG. 6 is a block diagram of the blockchain generation with ternary PUFs, according to one embodiment.

FIG. 7 is a block diagram of the blockchain generation with masked ternary PUFs, according to one embodiment.

FIG. 8 depicts a block diagram of the blockchain generation with ternary PUFs and a set of index, according to one embodiment.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Disclosed herein are multiple embodiments utilizing blockchain technology with binary PUFs and ternary PUFs.

DSA of Blockchains with Binary PUFs

According to one embodiment, the protocol includes the following steps:

1) Enrollment. The client devices securely download to the server arrays of PUF challenges that are stored in protected look-up tables (FIG.1). In this disclosure, "PUF challenges" are defined as initial measurements or a set of measurements of the PUF. For example, for a SRAM PUF, a challenge this will be a "0" or a "1" state for each cell after a power-off/power-on cycle. A variation could be to measure 100 times the SRAM PUF to statistically determine if a cell is more often a "0" than a "1". "PUF responses" are defined as the on-going measurements of the PUFs during the use of the device. A good PUF should be predictable, and the responses should be similar to the challenges. The

quantification of the quality of the PUF is given by the Hamming distance between challenges and responses, also defined as the challenge-response pair (CRP) error rates. The enrollment is done in a secure environment client by client. After enrollment, the constellation of clients are then operating in a non-secure environment and communicating with each other in a public network. Secure information needs to be encrypted.

2) Handshake. The handshake between the server and each client device has the objective to transmit the information needed to find a particular location within the PUF. Both the server and the client can independently find the same location in the look-up table and the PUF array. As described herein, an "Addressable PUF Generator" (APG) is defined as the device and methodology to point to a portion of the PUF. As shown in FIG. 2, the information needed can be a random number RN4j (RN4j = random number for client j and the fourth blockchain) feeding a hash function protected by a password. The message digest of the hash function is used to find an address in the PUF array. For example, if the array contains 256 rows and 256 columns, 8 bits of the message digest can be used to find the first coordinate X in the array; the following 8 bits can be used to find the second coordinate Y. As part of the handshake, the server sends the information needed to correct the responses and gets a low CRP error rate. This information, which is defined as the "helper", is the result of the error-correcting scheme of the overall architecture and an important piece of the protocol.

3) Public/Private Key Generation. Public and private keys are then generated from the PUF. The corrected response becomes the private keys while the public key is generated with an asymmetrical cryptographic algorithm such as ECC or RSA.

4) Generation of a Certified Blockchain. The blockchain is signed with DSA and the private key, using the asymmetrical cryptographic algorithm ECC, for example. Any observer can verify the validity of the transaction by decrypting the resulting cipher with the private key. The protocol allow one-time use of the public/private key pairs for the DSA.

5) Ledger. The server tracks the validity of the blockchains by independently generating a public/private key from the look-up tables. A ledger of all valid blockchains is made available to the peer-to-peer IoT infrastructure, thereby certifying the blockchains. As shown in FIG. 3, the ledger is organized by client and keeps track of the valid blockchains, their DSA, and the public key. This facilitates the trust between communicating peers and mitigates the risk associated with an illegitimate client gaining access to public/private key pairs.

One critical element of success of this protocol is the ability to fully correct the responses of the PUF to generate error-free private keys for the digital signature. A single-bit mismatch is not acceptable for public/private key encryption protocols. Typically, PUFs experience 3-10% CRP error rates when subjected to temperature changes, noise, aging, or parameter drift. The tradeoff is power consumption, latency, and complexity, as highly effective error-correcting methods trend to burden mainstream IoT devices.

DSA of Blockchains with Ternary PUFs

The objective of the novel architecture based PUF having ternary states is to reduce the CRP error rate while reducing computing power, latency, and failure rate due to poorly corrected PUF responses and without impacting entropy and the level of randomness of the PUFs. Assuming that the implementation should be ultra-low cost for IoTs, this method is easy to implement with commercially available components. According to one embodiment, the protocol with ternary PUFs includes the following steps:

1) Enrollment. As part of the enrollment, the cells of the entire PUF array of each client have to be sorted in three states,"-", "0", and "+". The cells with a ternary state "-"are at the bottom of the distribution and have stable binary challenge/response at "0". The cells with a ternary state "+" are at the top of the distribution and have stable binary challenge/response at "1". The cells with a ternary state "0" are at the middle of the distribution and are fuzzy. These fuzzy cells are ignored during challenge generation. For example, as shown in FIG. 4, the cells of a SRAM PUF need to be tested after successive power-off/power-on cycle. All cells that are showing instability are marked as fuzzy; the rest have either a low state or a high state. In the example shown in FIG. 4, 50 successive cycles with result in the identification of 10% to 40% of the cells as fuzzy. The CRP error rate of the remaining cells has been measured in the $10x^{-4}$ range, which is three orders of magnitude lower than what is measured with binary PUFs. In the case of Resistive RAM (ReRAM), the CRP error rate of ternary PUFs has been measured in the $10x^{-6}$ range.

As shown in FIG. 5, during enrollment, multiple readings of the PUFs are done in a secure environment client by client. The server stores the ternary challenges of their PUF in look-up tables by client. As described below, several methods are presented in this disclosure. In this section, it is assumed that the server sends the position of the fuzzy states in the PUF array back to each client. After enrollment, it is again assumed that the constellation of clients are operating in a non-secure environment and communicating with each other in a public network. Secure information has to be encrypted.

2) Handshake. The handshake between the server and each client device is exactly as presented previously with binary PUFs. The objective is to transmit the information needed to find a particular location within the PUF using the "helper" to correct the responses and get low CRP error rate. Both the server and the client can independently find the same location in the look-up table and the PUF array. As shown in FIG. 6, the server sends the random number RN4j and the helper 4j. The message digest of the hash function is again used to find an address in the PUF array. In deviation of the binary protocol above, both the client device and the server knows the position of the fuzzy cells and can independently skip these cells during CRP generation. The uncorrected CRP error rate is thereby reduced by several orders of magnitudes. As a result, the error-correcting scheme of such an architecture can be much lighter, consume less power, with short latency, and brief helper.

3) Public/Private Key Generation. Public and private keys are then generated from the PUF. The corrected responses become the private keys, while the public key is generated with an asymmetrical cryptographic algorithm such as ECC or RSA.

4) Generation of a Certified Blockchain. The blockchain is again signed with DSA and the private key using the asymmetrical cryptographic algorithm ECC, for example. As described previously with binary PUFs, any observer can verify the validity of the transaction by decrypting the resulting cipher with the private key. The protocol allows one-time use of the public/private key pairs for the DSA.

5) Ledger. The tracking of the validity of the blockchains is the same as described for the method employing binary PUFs.

The critical element of this protocol is the enrollment and the ability to sort out the ternary states of the PUF challenges. However, the enrollment is only done once; the subsequent generation of blockchains is much more efficient. One issue of this protocol, which is not necessarily a problem for most applications, is the need to store the position of the fuzzy states in the memory of each client device. The protocol presented below eliminates this issue.

Blockchain with Ternary PUFs and Masking Scheme
In this embodiment, the scheme is a variation of the ternary PUF protocol described above. The objective of this novel architecture is to eliminate the need to store the location of the fuzzy states in each client device. The protocol includes the following steps:

1) Enrollment. As described with the protocol employing ternary PUFs, the cells of the entire PUF array of each client are sorted in three states, "-", "0", and "+", with "0" representing the fuzzy cells. As shown in FIG. 5, the server stores the ternary challenges of their PUF store in look-up tables by client. In departure of the protocol described above, the server does not have to send the position of the fuzzy states in the PUF array back to each client. After enrollment, it is again assumed that the constellation of clients are operating in a non-secure environment, and communicating with each other in a public network. Secure information has to be encrypted.

2) Handshake. The handshake between the server and each client device needs to add "masking" information (FIG. 7). As described previously, the server send the random number RN4j and the helper 4j. The "mask" contains the positions of the fuzzy state that needs to be skipped after reading the responses generated by the addressable PUF array because they contain CRP errors. Both the client device and the server know the position of the fuzzy cells and can independently skip these cells during CRP generation. The uncorrected CRP error rate is then reduced by several orders of magnitudes; thus, the error-correcting scheme can be light, consume less power, operate with short latency, and use brief helper.

3) Public/Private Key Generation. Public and private keys are then generated from the PUF. The corrected responses become the private keys, while the public key is generated with an asymmetrical cryptographic algorithm such as ECC or RSA.

4) Generation of a Certified Blockchain. The blockchain is signed with DSA.

5) Ledger. The tracking of the validity of the blockchains is the same as described for the method employing binary PUFs.

In this protocol, the enrollment is also based on the ability to sort out the ternary states of the PUF challenges. There is no need to store the position of the fuzzy states in the memory of each client device. This simplifies the circuitry of the client device and eliminates the need to have secure memories in the client device. This also reduces the exposure to malicious entities if they take possession of the client device. For example, side-channel attacks detecting the position of the fuzzy cells during operation are no longer effective using this protocol.

Additional Protocols
The protocols above may be further modified by the methods described below.

Blockchain with Ternary PUFs and Indexing Scheme
This scheme is a variation of the protocol employing the masking scheme. As shown in FIG. 8, the objective of this novel architecture is to increase the level of confusion and mitigate exposure when the client devices are taken over by hacker.

Most PUFs can generate different fingerprints (i.e., CRPs) by using different methods of generation. For example, Resistive RAMs (ReRAMs) can generate CRPs by exploiting the various resistances of the cells subject to V_{Set} cycles or by exploiting the various resistance of the cells subject to V_{Reset} cycles. ReRAM can also generate CRPs with gate delays circuits, and with arbiter circuits.

In the scheme, the PUFs have k different ways to generate CRPs, k being a natural number. During enrollment each PUF generates a database of ternary challenges for each of these k ways with an associated index to recognize them. During the handshake, the server communicates the index to use for blockchain generation to the client device. All other sections of the protocols presented previously remain the same. This scheme increases entropy and offers an additional layer of security when the client devices are lost to the enemy and when subject to side-channel attacks.

Reduced Need to Use Error Correction Scheme
With ternary states, the likelihood of having a bad public/private key generation can be low. For example, if the PUF CRP error rate is 10^{-4}, the probability to have 256 bits error-free is within the 90% range. If the error rate is 10^{-6}, the probability is greater than 99%. Thereby, rather than using error correction, the server can verify the validity of the public/private key pair by checking the public key sent by the client. In case of a mismatch, the server can generate a stream of public/private key pairs with responses having a small Hamming distance with the challenge and determine if the public key generated by the client is acceptable.

For this case, the assumption is that the CRPs have 256 bits. The server can generate 255 streams from the challenges, then generate the associated 255 public keys, and compare them to the public key generated by the client device. The server will find a match if the Hamming distance between the challenge and the response generated by the server is one. If negative, the server can generate $\binom{256}{2}$ streams, which will be able to find a match if the Hamming distance is equal or

lower than two. Powerful servers can be used to authenticate greater error rates, if necessary. In case of a mismatch, the server can send a new random number and use a different portion of the APG. The ledger tracking the blockchains, as shown in FIG. 3, will then track the DSA and public keys generated by the client devices, but not the ones generated by the server.

This novel scheme is referred to as "public key matching"; this method simplifies the protocols and eliminates the need for error correction for ternary PUFs. This scheme is also applicable to binary PUFs when combined with error-correcting methods. Binary PUFs typically encounter 3-10% CRP error rates, so reducing the error rate to the sub 10^{-3} range is relatively light. However, reducing the CRP error rate below 10^{-8}, which is necessary for this type of application, is extremely complex. When combined with "public key matching", it is possible to use correcting methods reaching "only" sub-10^{-3} CRP error rates.

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 scheme is designed with arrays of addressable PUFs to digitally sign blockchains the following way:

1.1. The initial reading of the PUF, the challenges of a second device, are stored as reference by a first device, the server. The subsequent readings of the addressable PUF, the responses, are generated on demand by the addressable PUFs contained in the second device, the client device;

1.2. The first device send handshakes to the second device to identify portions of the addressable PUFs. From that portion, a stream of responses is extracted which is used to generate the private keys. From the private keys, and asymmetrical cryptography algorithm, public keys are generated which can decrypt ciphers encrypted by the private keys. The digital signatures of the blockchains are encrypted with the public keys, and communicated openly together with the public keys;

1.3. The first device independently extract initial private keys from the challenges of the addressable PUFs, and generate public keys with asymmetrical cryptography algorithm;

To verify that the public keys generated by the second device are valid, the first device compares the public keys independently generated by each device. If the keys are not matching, the first device generate a stream of private keys having small Hamming distance with the initial private keys, and associated public keys. If one public key is matching with the one generated by the second device, the digital signature of the blockchain is considered as valid by the first device.

2. A scheme is designed with arrays of ternary addressable PUFs to digitally sign blockchains the following way:

2.1 During the initial characterization of the ternary PUFs of a second device, the cells are sorted into three categories of challenges: the ones that are solidly on the low side, the "-"s, the ones that are solidly on the high side, the "+"s, the others that are in the middle or fuzzy, the "0"s. The challenges are stored as reference by a first device, the server. The positions of the fuzzy cells are communicated to the second device by the first device, which store the information. The subsequent readings of the ternary addressable PUF, the responses, are generated on demand by the addressable PUFs contained in the second device while skipping the known fuzzy cells;

2.2 The first device send handshakes to the second device to identify portions of the addressable PUFs. From that portion, a stream of responses is extracted while skipping the fuzzy cells, which is used to generate the private keys. From the private keys, and asymmetrical cryptography algorithm, public keys are generated which can decrypt ciphers encrypted by the private keys. The digital signatures of the blockchains are encrypted with the public keys, and communicated openly together with the public keys;

2.3 The first device independently extract initial private keys from the challenges of the addressable PUFs, while skipping the fuzzy cells, and generate public keys with asymmetrical cryptography algorithm. To verify that the public keys generated by the second device are valid, the first device compares the public keys independently generated by each device. If the keys are not matching, the first device generate a stream of private keys having small Hamming distance with the initial private keys, and associated public keys. If one public key is matching with the

one generated by the second device, the digital signature of the blockchain is considered as valid by the first device.

3. A scheme is designed with arrays of ternary addressable PUFs to digitally sign blockchains the following way:

3.1 During the initial characterization of the ternary PUFs of a second device, the cells are sorted into three categories of challenges: the ones that are solidly on the low side, the "-"s, the ones that are solidly on the high side, the "+"s, the others that are in the middle or fuzzy, the "O"s. The challenges are stored as reference by a first device, the server. The subsequent readings of the ternary addressable PUF, the responses, are generated on demand by the addressable PUFs contained in the second device;

3.2 The first device send handshakes to the second device to identify portions of the addressable PUFs, as well as a mask to identify the position of the fuzzy cells. From that portion, a stream of responses is extracted while skipping the fuzzy cells, which is used to generate the private keys. From the private keys, and asymmetrical cryptography algorithm, public keys are generated which can decrypt ciphers encrypted by the private keys. The digital signatures of the blockchains are encrypted with the public keys, and communicated openly together with the public keys;

3.3 The first device independently extract initial private keys from the challenges of the addressable PUFs, while skipping the fuzzy cells, and generate public keys with asymmetrical cryptography algorithm. To verify that the public keys generated by the second device are valid, the first device compares the public keys independently generated by each device. If the keys are not matching, the first device generate a stream of private keys having small Hamming distance with the initial private keys, and associated public keys. If one public key is matching with the one generated by the second device, the digital signature of the blockchain is considered as valid by the first device.

4. Wherein error-correcting methods are used in claims 1 to 3 to improve the matching between challenges and responses, and reduce error rates.

5. Wherein hashing functions and multifactor authentication are used to protect the information communicated between the first and the second device during handshake in claims 1 to 3.

6. Wherein claims 1 to 3 use RSA, ECC, ECC-GF, or DH as asymmetrical encryption algorithm.

7. Wherein the physical unclonable functions of claims 1 to 3 are based on gate delays, arbiter, ring oscillators, TCAM, S RAM, Flash memory, D RAM, Resistive RAM, Conductive Bridge RAM, M RAM, memristors, Ferro RAM, EEPROM, PROM, optical PUF, or sensor based PUF.

8. Wherein an index methodology is used to enhance entropy by creating multiple challenge-response pairs from the same PUF.

ABSTRACT

Physical Unclonable functions (PUFs) can generate electronic fingerprints from the components of the client devices of an Internet of Things (IoT) infrastructure. This invention describes ways to generate reliable public/private key pairs from the PUF fingerprints that can be used for the digital signature algorithm (DSA) to authenticate blockchains generated by the client's devices. The server driving the IoT infrastructure can independently generate the same public/private key pairs and certify that the client device used valid public keys. The positions of the fingerprint that are fuzzy are known by the server and tracked with a ternary state. This knowledge allows reductions in the error rate during public/private key generation and provides additional layers of security.

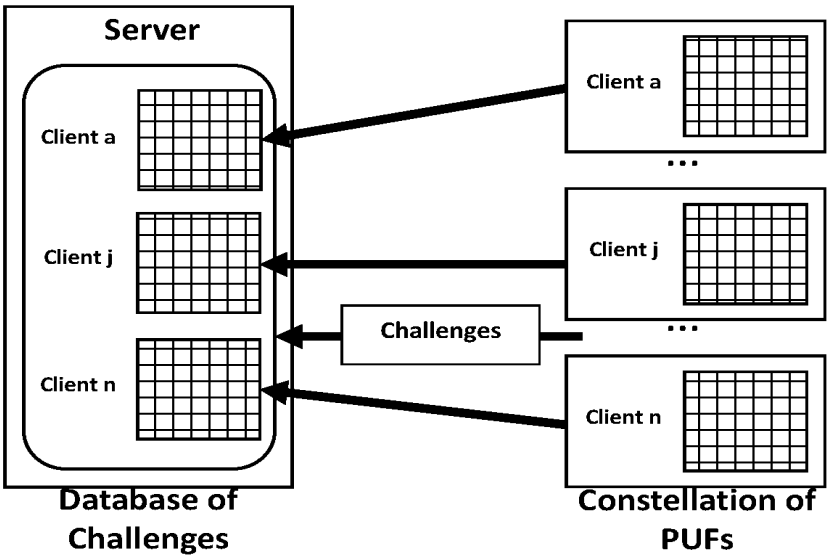
Server
Client a
Client j
Client n
Database of Challenges
Challenges
Client a
Client j
Client n
Constellation of PUFs

FIG. 1

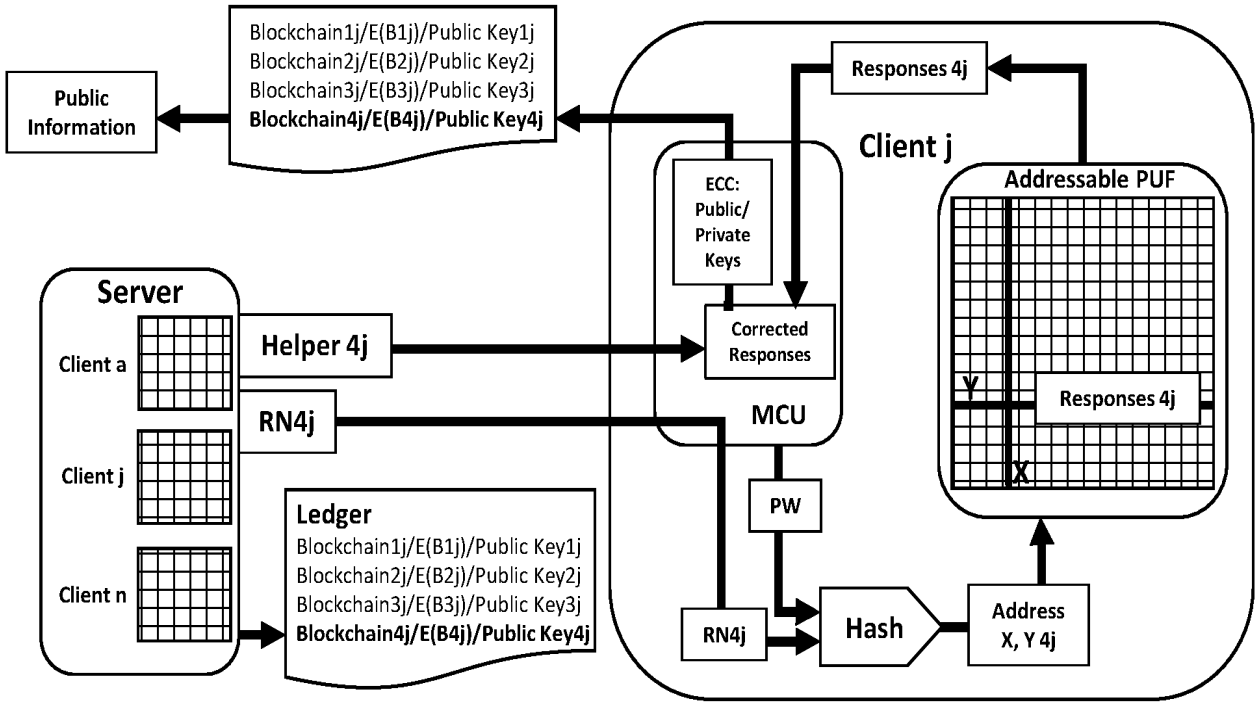
Blockchain1j/E(B1j)/Public Key1j
Blockchain2j/E(B2j)/Public Key2j
Blockchain3j/E(B3j)/Public Key3j
Blockchain4j/E(B4j)/Public Key4j
Public Information
Responses 4j
Client j
ECC: Public/ Private Keys
Addressable PUF
Server
Client a
Client j
Client n
Helper 4j
RN4j
Corrected Responses
MCU
Responses 4j
Y
X
PW
Ledger
Blockchain1j/E(B1j)/Public Key1j
Blockchain2j/E(B2j)/Public Key2j
Blockchain3j/E(B3j)/Public Key3j
Blockchain4j/E(B4j)/Public Key4j
RN4j
Hash
Address X, Y 4j

FIG. 2

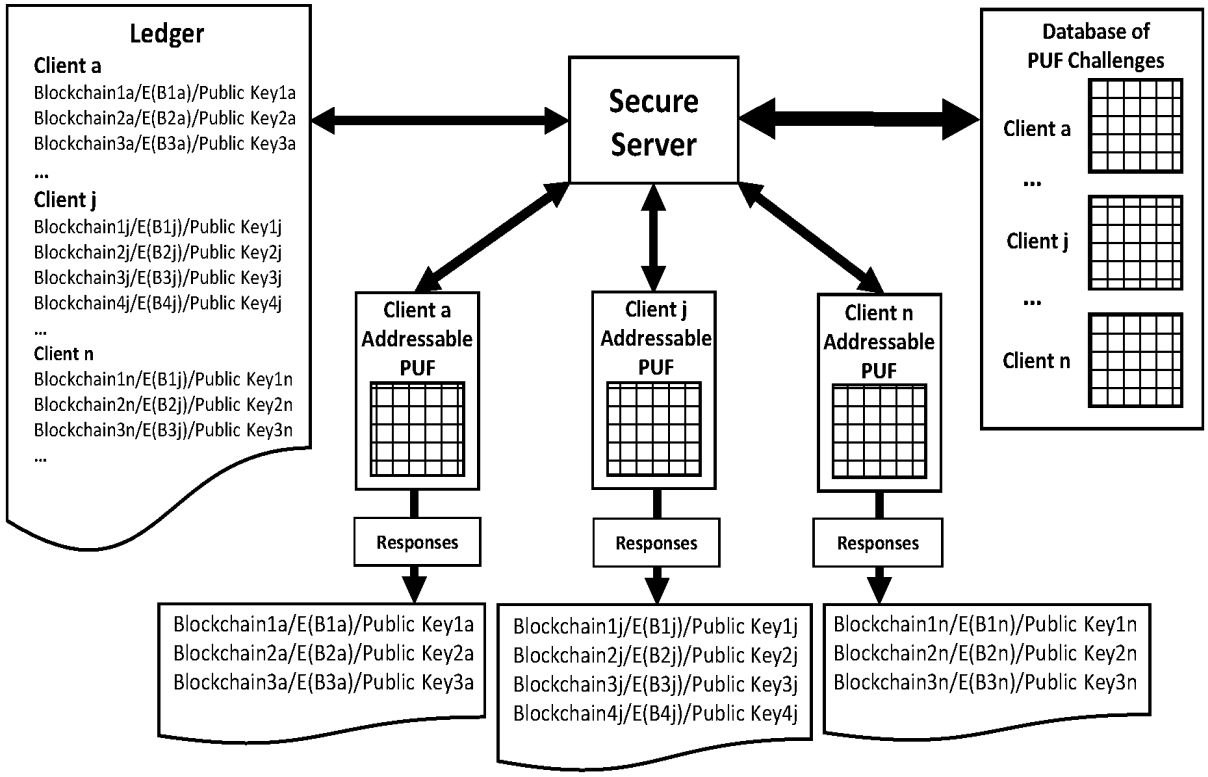
Ledger
Client a
Blockchain1a/E(B1a)/Public Key1a
Blockchain2a/E(B2a)/Public Key2a
Blockchain3a/E(B3a)/Public Key3a
...
Client j
Blockchain1j/E(B1j)/Public Key1j
Blockchain2j/E(B2j)/Public Key2j
Blockchain3j/E(B3j)/Public Key3j
Blockchain4j/E(B4j)/Public Key4j
...
Client n
Blockchain1n/E(B1j)/Public Key1n
Blockchain2n/E(B2j)/Public Key2n
Blockchain3n/E(B3j)/Public Key3n
...
Secure Server
Database of PUF Challenges
Client a
...
Client j
...
Client n
Client a Addressable PUF
Client j Addressable PUF
Client n Addressable PUF
Responses
Responses
Responses
Blockchain1a/E(B1a)/Public Key1a
Blockchain2a/E(B2a)/Public Key2a
Blockchain3a/E(B3a)/Public Key3a
Blockchain1j/E(B1j)/Public Key1j
Blockchain2j/E(B2j)/Public Key2j
Blockchain3j/E(B3j)/Public Key3j
Blockchain4j/E(B4j)/Public Key4j
Blockchain1n/E(B1n)/Public Key1n
Blockchain2n/E(B2n)/Public Key2n
Blockchain3n/E(B3n)/Public Key3n

FIG. 3

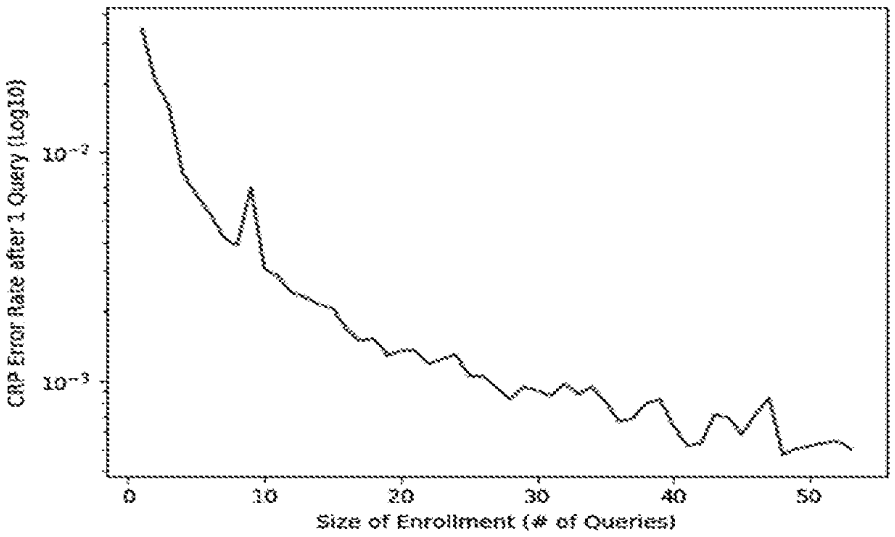
CRP Error Rate after 1 Query (Log10)
10^{-2}
10^{-3}
0
10
20
30
40
50
Size of Enrollment (# of Queries)

FIG. 4

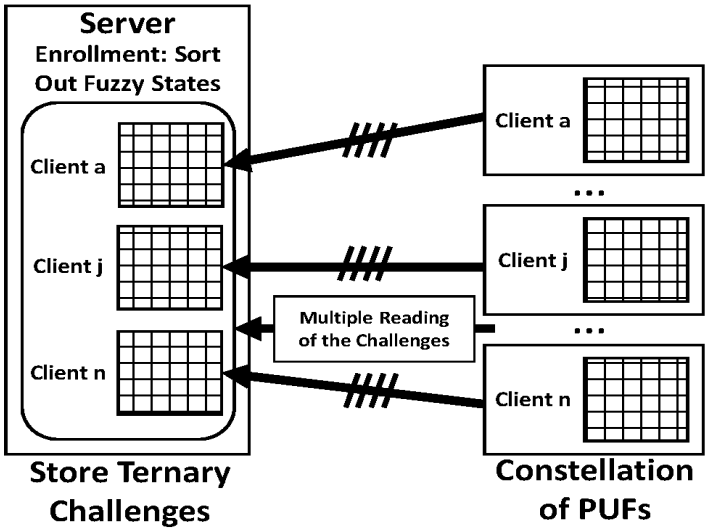
Server
Enrollment: Sort Out Fuzzy States
Client a
Client j
Client n
Store Ternary Challenges
Multiple Reading of the Challenges
Client a
Client j
Client n
Constellation of PUFs

FIG. 5

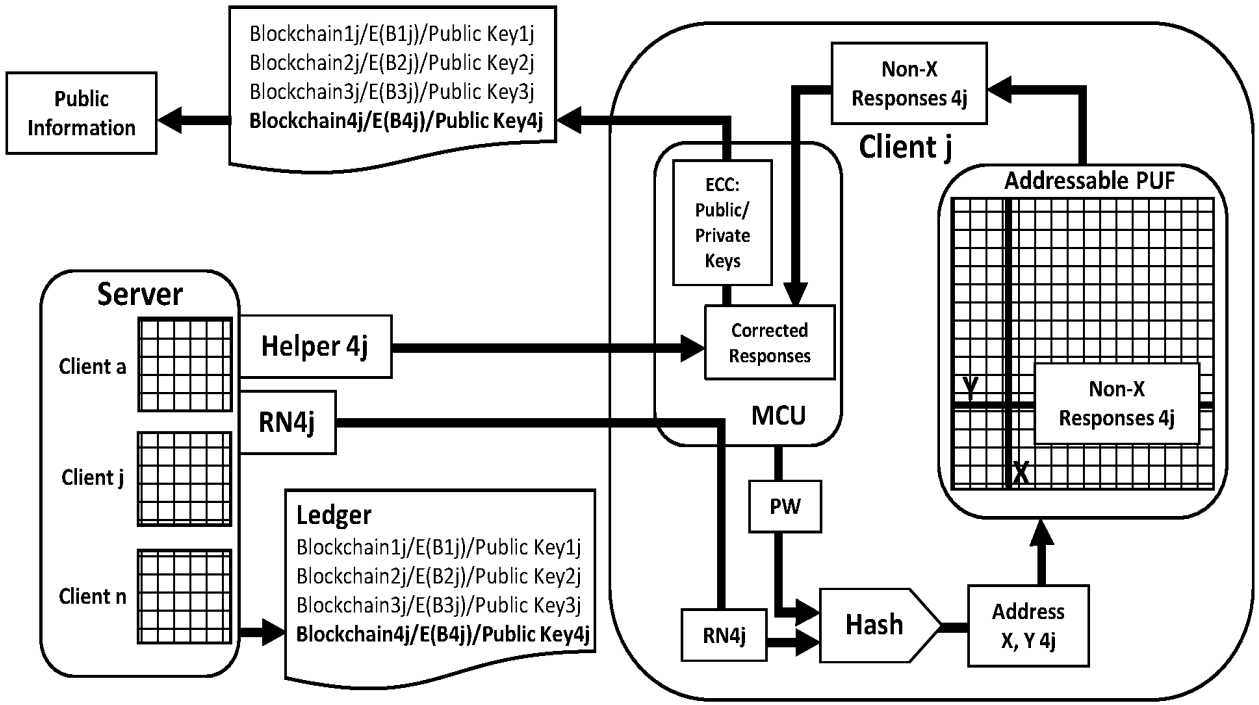
Blockchain1j/E(B1j)/Public Key1j
Blockchain2j/E(B2j)/Public Key2j
Blockchain3j/E(B3j)/Public Key3j
Blockchain4j/E(B4j)/Public Key4j
Public Information
Non-X Responses 4j
Client j
ECC: Public/ Private Keys
Addressable PUF
Server
Client a
Client j
Client n
Helper 4j
RN4j
Corrected Responses
MCU
Y
X
Non-X Responses 4j
PW
Ledger
Blockchain1j/E(B1j)/Public Key1j
Blockchain2j/E(B2j)/Public Key2j
Blockchain3j/E(B3j)/Public Key3j
Blockchain4j/E(B4j)/Public Key4j
RN4j
Hash
Address X, Y 4j

FIG. 6

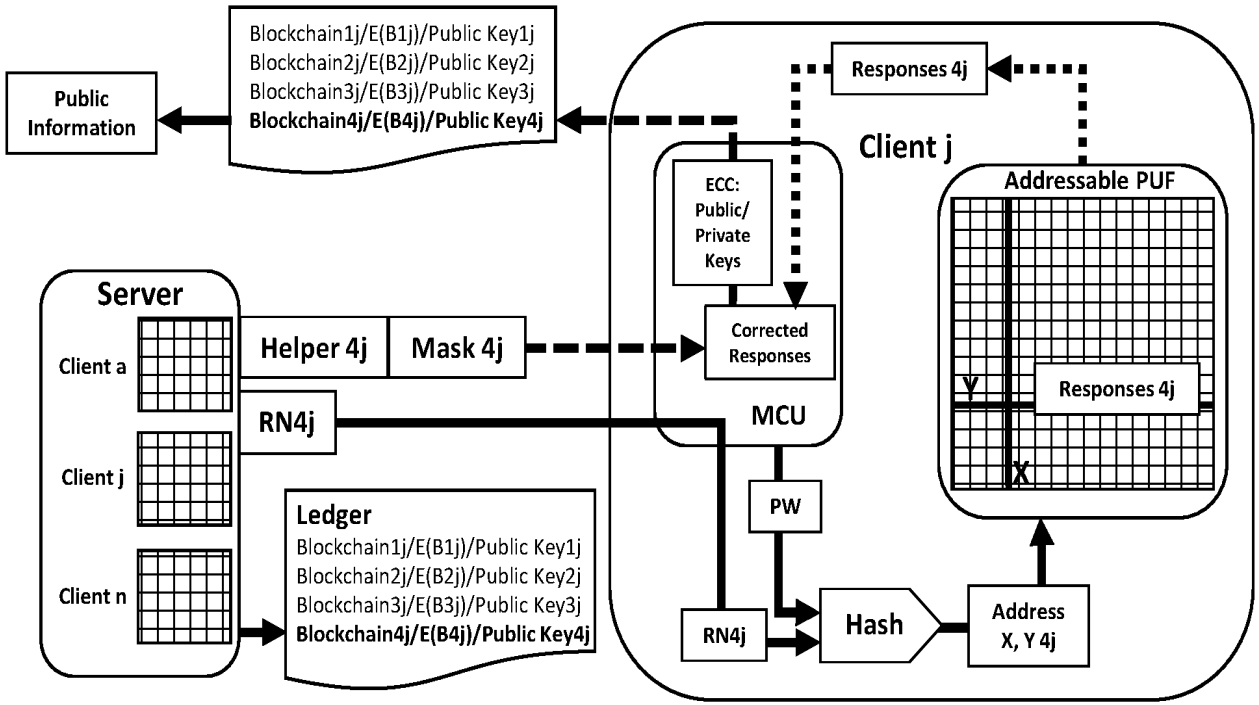
Blockchain1j/E(B1j)/Public Key1j
Blockchain2j/E(B2j)/Public Key2j
Blockchain3j/E(B3j)/Public Key3j
Blockchain4j/E(B4j)/Public Key4j
Public Information
Responses 4j
Client j
ECC: Public/ Private Keys
Addressable PUF
Server
Client a
Client j
Client n
Helper 4j
Mask 4j
RN4j
Corrected Responses
MCU
Y
X
Responses 4j
PW
Ledger
Blockchain1j/E(B1j)/Public Key1j
Blockchain2j/E(B2j)/Public Key2j
Blockchain3j/E(B3j)/Public Key3j
Blockchain4j/E(B4j)/Public Key4j
RN4j
Hash
Address X, Y 4j

FIG. 7

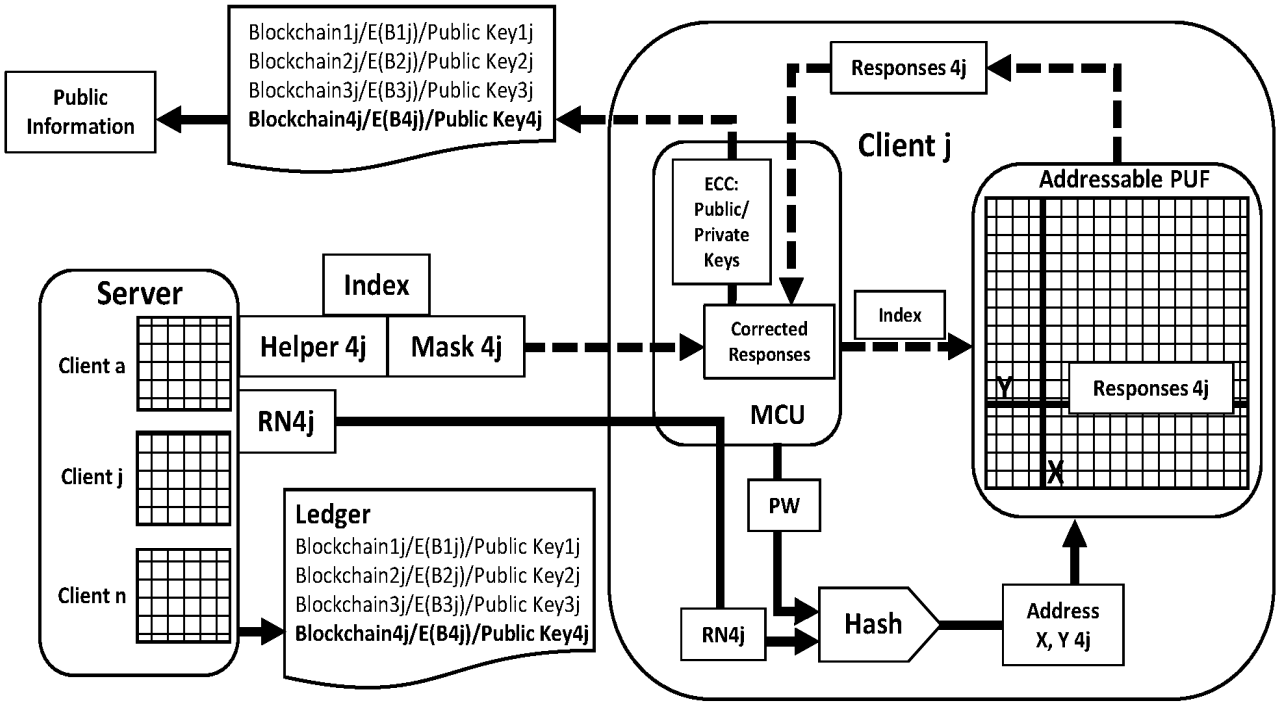
Blockchain1j/E(B1j)/Public Key1j
Blockchain2j/E(B2j)/Public Key2j
Blockchain3j/E(B3j)/Public Key3j
Blockchain4j/E(B4j)/Public Key4j
Public Information
Responses 4j
Client j
ECC: Public/ Private Keys
Addressable PUF
Server
Index
Client a
Helper 4j
Mask 4j
Corrected Responses
Index
Y
Responses 4j
RN4j
MCU
Client j
X
PW
Ledger
Blockchain1j/E(B1j)/Public Key1j
Blockchain2j/E(B2j)/Public Key2j
Blockchain3j/E(B3j)/Public Key3j
Blockchain4j/E(B4j)/Public Key4j
Client n
RN4j
Hash
Address X, Y 4j

FIG. 8