Blockchain-based Automated Certificate Revocation for 5G IoT

Tharaka Hewa*, An Braeken†, Mika Ylianttila‡, Madhusanka Liyanage§
*‡Centre for Wireless Communications, University of Oulu, Finland
†Vrije Universiteit Brussel, Brussels, Belgium
§School of Computer Science, University College Dublin, Ireland
Email: *‡§[firstname.lastname]@oulu.fi, †an.braeken@vub.be §madhusanka@ucd.ie

***Abstract*—Internet of Things (IoT) is a key topic of interest in modern communication context with the evolution of 5G and beyond ecosystems. 5G will interconnects billions of IoT devices wirelessly. The wireless communication exposes the devices to massive security risks in different dimensions. The Public Key Infrastructure (PKI) is one of the promising solutions to eliminate security risks. It ensures the authentication and communication integrity by using public key certificates. However, the overhead of certificate storage is a significant problem for the resource constrained IoT devices. We propose an application of Elliptic Curve Qu Vanstone (ECQV) certificates, which are lightweight in size for the resource restricted IoT devices. Furthermore, we incorporate the blockchain based smart contracts to handle the certificate related operations. We utilize the smart contracts in the certificate issuance and developed a smart contract based threat scoring mechanism to automatically revoke the certificates. The lightweight nature of ECQV certificates enables the distributed ledger to store, update, and revoke the certificates. We evaluated the proposed solution in Hyperledger Fabric blockchain platform.**

***Index Terms*—Elliptic Curve Cryptography, Qu Vanstone Certificates, Blockchain, Smart Contracts, 5G, IoT**

I. Introduction

Internet of Things (IoT) allows the interconnection of billions or even trillions of objects through the internet and is growing at tremendous rate. IoT basically represent a network of physical objects or devices, consisting of sensors and actuators, enabling a multitude of applications and services by exchanging data with each other and the end user. This requires compute intensive operations, huge storage needs and realtime communication, which can not always be guaranteed in the most efficient way by cloud service providers. Therefore, fog computing has been introduced, in which fog devices perform the first data processing activities, resulting in a significant reduction of delay for the application.

As most of the communications are performed in a wireless medium, open for a wide range of attackers, the inclusion of sufficient security mechanisms should be guaranteed [1]. In particular, authentication of legitimate IoT devices is a very important feature [2]. Authentication is typically addressed by means of certificates, issued by a particular Certificate Authority (CA). In particular, in case of IoT, the Elliptic Curve Qu-Van Stone certificates offer a lightweight solution. However, as IoT devices are put in open field, they can be more easily attacked and hijacked. As a consequence, it should be possible to organise the revocation of certificates, issued by multiple CAs, in an efficient way. This is typically done by consulting a Certificate Revocation List (CRL), which requires a lot of storage memory, is time consuming and not easily manageable in case different CAs are involved.

A comprehensive discussion on the role of blockchain in the 5G and IoT with opportunities and challenges discussed in [3]. The goal of this paper is to investigate the feasibility of using distributed ledger technologies to organise the authentication of IoT devices in an efficient way for a fog computing architecture. The IoT devices should be able to contact the fog devices in an authenticated way using their certificates published on the distributed ledger by their CA. The fog is in the possibility to easily verify the validity of the certificates by consulting the ledger. The servers, responsible for the fog architecture are able to monitor the traffic and to determine potentially malicious devices (e.g. through intrusion detection mechanisms). If so, the server revokes the corresponding certificate of the device and publishes the revocation on the ledger.

We utilized the CA with a private blockchain platform to enable Elliptic Curve Qu Vansone (ECQV) certificate generation, access, and revocation using the smart contracts. We compared the impact of time and memory of applying the blockchain on each operation with scaling up the number of requests. The blockchain service provides Application Programming Interface(API) to receive threat related events from the trusted threat acknowledgement services within the network and assigns a threat score for the threat. When the threat score exceeded, the smart contract automatically revokes the certificate. We validated the solution with simulation and elaborated with an implementation to evaluate the impact of memory and time parameters as well as the scalability on each type of operation.

II. Background

A. Elliptic Curve based operations

The most lightweight public key cryptographic solutions can be organised by means of Elliptic Curve Cryptography (ECC).

ECC is based on the algebraic structure of ECs over finite fields. The curve in the finite field F_p is denoted by $E_{p(a,b)}$ and is defined by the equation $y^2 = x^3 + ax + b$, where a

and b are two constants in F_p and $\Delta = 4a^3 + 27b^2 \neq 0$. The base point generator of $E_{p(a,b)}$ of prime order q is denoted by G. All points on $E_{p(a,b)}$, together with the infinite point form an additive group. The curve E25519 [5] in F_p with $p = 2^{255} - 19$ represents currently one of the fastest curves, possessing in addition also resistance against some well-known implementation attacks [4]. As the EC points are compressed, it is sufficient to use the X-coordinate for the representation of the complete point.

There are two main operations in ECC, being addition and multiplication. The addition of two points, $P1 + P2 = R$, results in a new EC point R. The scalar EC multiplication with $r \in F_q$ for a given EC point P is represented by $R = rP = (R_x, R_y)$, with $R_x, R_y \in F_p$, resulting in the point R of the EC.

The concatenation and xor operation of two messages M_1 and M_2 is denoted by $M_1 \| M_2$ and $M_1 \oplus M_2$ respectively. In addition, the $SHAKE128(M, d)$ algorithm is used, which is the $Keccak[256](M \| 1111 \| d)$ scheme applied on message M with output size d. This scheme offers a security strength of $\min(d/2, 128)$ bits against collision and pre-image attacks and $\min(d, 128)$ bits against 2nd pre-image attacks [6]. The default mode $SHAKE128(M, 256)$ can be used as a hash function with a 256-bit length and 128-bit overall security. The notation H is used to refer to this function.

We now describe the ECQV mechanism, used to define the certificates proposed in our setting.

1) Elliptic Curve Qu Vanstone Mechanisms: The Certificate Authority (CA) is responsible for this process and is considered to be a trusted entity. The identity ID_{CA} and corresponding public key Q_{CA} should be publicly available. In our context, this would correspond to the publication of both parameters on the ledger. The corresponding private key d_{CA} is kept secret at the CA.

The different steps in the derivation of the key pair (d_n, Q_n) for an entity with identity ID_n are as follows. First, the device needs to send its identity and an EC point $R_n = r_n G$, with r_n randomly chosen. Based on this information, the CA performs the following operations.

- First, the CA chooses its own random value $r_{CA} \in F_q^*$ and computes $R_{CA} = r_{CA} P$. Then the certificate $cert_n$ is defined by $cert_n = R_{CA} + R_n$.
- The value $r = H(cert_n \| ID_n) r_{CA} + d_{CA}$ is computed.
- The tuple $(cert_n, r)$ is sent to the user.

The device can then derive its private key by $d_n = H(cert_n \| ID_n) r_n + r$ and the corresponding public key equals to $P_n = d_n P$.

Note that any other device/user can derive the public key of the entity with identity ID_n given its certificate $cert_n$ by

$$Q_n = H(cert_n \| ID_n) cert_n + Q_{CA}.$$

B. Blockchain and the Distributed Ledger Technology

Blockchain and the Distributed Ledger Technology (DLT) is one of the most hot topics which attracted strong interest by the industry and the academia. The distributed nature of blockchain and the autonomous accurate execution of smart contracts open broader scope of innovations. Especially, in the telecommunication context, the distributed nature of blockchain and smart contracts are advantageous in terms of efficiency, security and so on. The distributed ledger in blockchain is a versatile application which eliminates the centralization of data storage along with the guaranteed integrity of data storage.

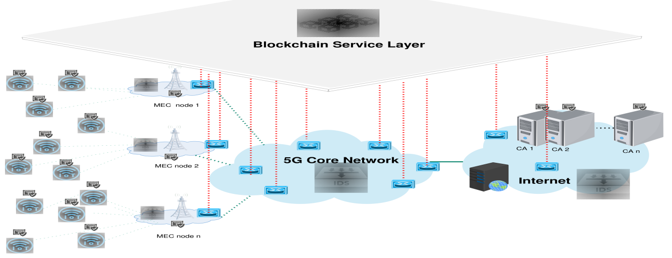
III. Related Work

The ECQV mechanism [7] possesses several interesting properties. First, the entity requesting for the security material, is the only one, thus also not the CA, able to derive the private key. Therefore, the scheme is secure against key escrow. Second, no secure channel is required during the process as all variables can be sent over the open channel. Only an initial check on the validity of the identity is required. Finally, the scheme only requires the identity (e.g. 32-bit variable) and certificate (point of the curve), in order to derive the corresponding public key. This principle of implicit certificates leads to very small size of the certificates, compared to other approaches like X.509 certificates. The ECQV mechanism is typically applied in IoT contexts [8], [9], which is also a domain characterized by its constrained nature. A concrete applications of ECQV implicit certificates for the vehicular networks is presented in [10], [11] utilizing the advantage of smaller size of the certificate. Ethereum is the first smart contract based innovation which is explained in [17]. The public blockchain application to the PKI context is presented in [12]–[16]. The scaling up Hyperledger Fabric blockchain network is explained in [18].

IV. Proposed Architecture

For the communication between IoT and fog device, a lot of well established authentication and key agreement algorithms exist. Therefore, we here consider that the fog received both the identity and certificate, established by means of the ECQV mechanism, and is able to derive the corresponding public key of the device. The focus of our system will be the verification of the validity of the public key or certificate at the distributed ledger, taking into account that potential revocations have been added to the ledger by servers. The miners consist of servers and active CAs. The significant modules of the proposed

Fig. 1. The proposed architecture



architecture are as follows.

A. Blockchain Service Layer

The blockchain service layer plays a vital role by acting as a service provider in the proposed architecture providing transparency, validation of the certificate request and accuracy in the threat scoring and revoking operation. There are a few smart contracts deployed to operate on different activities of the certificate life cycle. The blockchain is working separately and the main objective is to adopt to Blockchain as a Service (BaaS) architecture in the system.

B. Certification Authority(CA)

The proposed solution makes identity ID_{CA} and corresponding public key Q_{CA} publicly available. The corresponding private key d_{CA} is kept secret at the CA. Furthermore, the CA publishes the CRL, which consists of the revoked certificates. The CA only receives the certificate generation requests upon the basic validation of attributes from the smart contract. The smart contract submits the certificate generation request to the CA after a basic validation through the smart contract.

C. Certificate Access Nodes

The certificate access node is a high level reference to any node of the IoT ecosystem. In the PKI ecosystem perspective, there are three broad categories identified, segregated by the interaction within the network.

1) Certificate subscribing nodes: The subscriber node selects an EC point $R_n = r_n G$ and submits to the blockchain service along with the identity information. The CA extracts the identity information and proceeds with the certificate generation. The $cert_n$ and value r generated as the tuple $(cert_n, r)$ is available in the ledger enabling the subscriber to derive its corresponding private key and complete the certificate generation process.

2) Certificate status querying nodes: The nodes in the ecosystem require to check the certificate status prior to a particular operation. The proposed solution provides a smart contract based threat scoring mechanism through the blockchain service layer. In addition to that, the smart contracts are applicable to establish the granularity of access to the certificate according to the node category.

3) Threat acknowledging nodes: The threat alerting smart contracts update the threat score according to the alert information received via services such as Intrusion Detection Systems (IDS).

D. Blockchain based smart contracts

The smart contracts are an immutable programs deployed in the nodes of the PKI ecosystem to execute when the certain conditions are met. The smart contracts utilized the subscriber on-boarding and attack notification. The smart contracts guarantee accurate and autonomous conditional execution on each operation. It is important to ensure the immutability in the threat scoring mechanism to guarantee the accuracy and alterations in the scoring.

1) Certificate issuance smart contract: The certificate issuance smart contract is invoked when a new IoT device requests a certificate. The smart contract performs a primary validation to eliminate the sybil attacks before sending the request to the CA. The certificate issuance smart contract invokes the certificate issuance request via calling off-chain the CA service via the smart contract. The smart contract also updates the ledger including the corresponding information.

Algorithm 1 Threat scoring and revocation smart contract

```
Require: $attackType$, $certIdentifer$
  $certificate \leftarrow$getCertificateFromLedger($certIdentifer$)
  if $attackType == attackType1$ then
    $certificate.threatScore \leftarrow certificate.threatScore + score1$
  else
    $certificate.threatScore \leftarrow certificate.threatScore + score2$
  end if
  if $certificate.threatScore > threashold$ then
    $certificate.status \leftarrow revoked$
  end if
  updateCertificateInLedger($certificate$)
```

2) Threat scoring and revocation smart contract: The threat scoring smart contract is encoded with the scoring algorithm as illustrated in Algorithm 1. In a more specific implementation, the type of attacks and the corresponding scores can be defined in the smart contracts. For instance, for ICMP traffic, the threat score is $score1$ while for the portscan, the threat score is $score2$. Each attack information is acknowledged by the integrated IDS of the system. The threshold value to revoke the certificate also can be defined in the smart contract. The main advantage of the mechanism is the accurate execution of the pre-defined conditions in a transparent way, which leads to increment the threat score and eventually revoke the certificate. The smart contract retrieves the certificate object for each event and updates the threat score of the device associated with the certificate. When the threat score exceeds the threshold level of threat score to revoke the certificate, the certificate will be marked as a revoked certificate and acknowledge the ledger. The events corresponding to the certificate revocation are logged in the ledger and transparently available for the post investigations.

V. Numerical Results

The key objective of this section is to validate the proposed architecture and investigate the impact of time and memory in the CA related operations on the simulation environment. The principal requirement of the proposed architecture is to enable the resource restricted IoT devices to interact with the certificate related operations outperforming the features of the existing CA based architecture. Therefore, time, memory consumption and scalability need to be carefully evaluated.

A. Simulation environment

The computing infrastructure consists of one virtual machine and one host machine. The virtual machine runs Ubuntu 18.10 64 bit with 13.2GB RAM and single core allocation. The host machine consists of Intel(R) Core i5 -8250 CPU with four cores and eight logical processors. The Hyperledger Fabric blockchain deployed on the virtual machine is a single order mode with REST API connectivity.

B. Assumptions

1) The communication channel between IoT device and blockchain service is secured.
2) The threat acknowledgement messages are received from a trusted threat detector.
3) The communication channel between the threat detector and the blockchain service is secured.
4) The randomly generated certificate serial number does not correspond to a previously issued certificate.
5) The randomly generated certificate serial number does not correspond to a previously issued certificate.

C. Transaction generation

The transactions follow a Poisson arrival with defined mean values corresponding to the transactions per second (tps). The rate parameter λ is defined for the generated transactions per second. The simulation of IoT device is performed by the multi-threaded software codes in simulation. The rate parameter is configurable when the test is conducted. The transaction generation follows a Poisson distribution with rate parameter λ and the probability of observing k events in the time period denoted as

$$P(X = k) = \frac{e^{-\lambda}\lambda^k}{k!}$$

The average of the measuring parameter R for N requests triggered on each λ is denoted as

$$\bar{R} = \frac{1}{N}\sum_{i=1}^{N} R_i$$

The runtime memory consumption and transaction time such as elapsed time to the round trip of a function call are the main parameters measured in the evaluation.

D. Core components of the simulation setup

1) Certification Authority: The CA simulates the state of art CA in a PKI ecosystem. The CA is implemented using Java programming language utilizing cryptographic libraries such as BouncyCastle cryptographic library. The CA supports signature algorithms based on curves such as NIST P-256, Brainpool 256 r1, and Brainpool 384 r1. The CA is deployed with REST API to receive the certificate generation requests from the smart contract. The ElasticSearch, which is the Lucene based state of art optimal search engine, is connected as the backend for the data storage with anticipation of the optimal retrieval.

2) Blockchain and smart contracts: We use Hyperledger Fabric private blockchain in the blockchain service layer. The smart contracts are encoded in the Javascript and capable in off-chain invocation of the CA for the certificate generation. The blockchain service layer consists of significant services including order, storage, peer and Hyperledger CA service (for Hyperledger associated certificate management) deployed as Docker containers. However, there is no restriction of the blockchain platform. The current implementation utilizes Hyperledger Fabric as the blockchain platform. The Hyperledger Fabric blockchain platform deployed with API to be invoked by the different services including IDS, and CA.

3) Certificate accessing IoT nodes: The IoT nodes perform certificate generation, certificate retrieval, and threat acknowledgement requests to either the blockchain service layer or the CA. The IoT nodes simulated using the software codes are developed using the REST API clients and are capable of measuring the elapsed time in milliseconds for the operations and run-time memory consumption on the invocation of REST API calls for each operation.

E. Simulation

1) Certificate generation: The certificate generation is focused on the behavior of the certificate generation time with the developed state of art CA simulator and with the proposed solution. Furthermore, to evaluate the behavior when scaling the proposed solution, the number of threads is increased with corresponding values aligning with the Poisson arrival process. The elapsed time from the point of submission of identity and required parameters to the point of generation of the implicit certificate is measured. The certificate generation is generally considered as a once a lifetime operation in a PKI ecosystem.

2) Certificate access for verification/status check: The certificate access operation is one of the most important and frequent actions on the PKI ecosystem. The nodes may need to check the certificate status before connecting or interacting with the nodes which hold the certificates. Furthermore, the proposed solution reflects the threat score, which corresponds to the state of the certificate with attack history within the network. The run-time memory consumption of the function calls and the elapsed time for the round trip of the query are evaluated in the experiment.

3) Threat scoring and certificate revocation: Each attack type assigned a score and a threshold value encoded in the smart contract. The threat score is updated after the attack notification and once the threat score exceeds, the smart contract automatically revokes the certificate. The revocation reasons and attack history of a particular certificate are logged and published in the ledger for further investigations. Software codes are used to simulate the attacking alerts to update the blockchain. A concrete implementation of the threat scoring and certificate revocation algorithm (Algorithm 1) is implemented in the smart contract. Depending on the attack type acknowledged by the Suricata IDS, the threat score is updated. The attack types in the smart contracts include Internet Control

Message Protocol (ICMP) traffic, port-scan attempts and key compromise.

F. Simulation Results

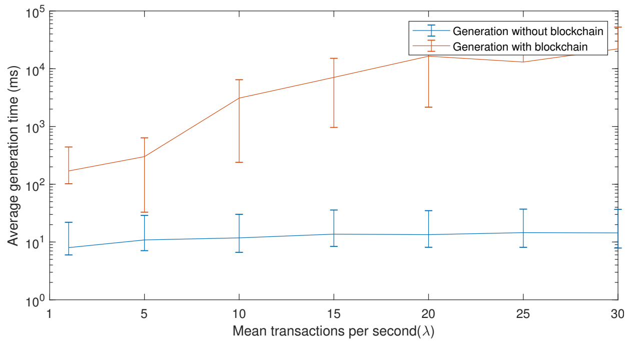


Fig. 2. ECQV implicit certificate generation elapsed time with and without blockchain

1) Certificate generation: The simulation for the certificate generation is performed using a multi-threaded program, which generates transaction traffic in Poisson arrival according to each lambda value. The mean transaction is scaled from 1 to 30 per second and the results are displayed in Figure 2. We observed that there is a limitation in scaling the transaction rate due to the simulation deployment model of the Hyperledger Fabric blockchain. A scaled deployment model will eliminate the scaling limitations in the single order deployment model. However the certificate generation is considered as a lifetime request in a PKI ecosystem because a device will request a certificate once in a lifetime or upon expiry.

2) Certificate access for verification/status check: Certificate access is considered as a key operation on the PKI ecosystem. The impact of runtime memory and elapsed time for the query are evaluated and represented in Figure 3 (a) and (b). The blockchain outperforms the certificate access query with ElasticSearch based CA simulator in terms of memory and time efficiency. This is an anticipated behavior of the proposed architecture since accessing the certificate from the blockchain is similar to accessing a database directly. Even though ElasticSearch is an optimal data storage for searching, it cannot be interfaced for access to the accessing nodes due to security reasons. ElasticSearch interfaces with the nodes using a requirement specific API such as RESTful services to limit the scope of access of the subscribers for data. In contrast, the blockchain is accessible as a database directly and with the built-in consensus and integrity mechanisms, the blockchain is safe from attacks such as data alterations and DoS attacks. The lightweight nature of the ECQV implicit certificate takes advantage of storing on the blockchain with minimal storage overhead and faster querying capability. The simulation scaled up to 500 transactions per second and are even more scalable for querying on the blockchain to access certificates.

3) Threat scoring and certificate revocation: Elapsed time and memory for the certificate threat acknowledgement and certificate revocation are illustrated in Figure 4 (a) and (b).

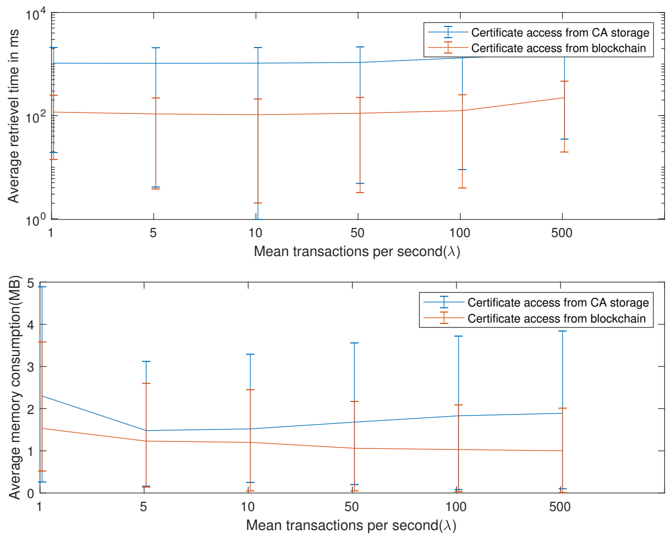


Fig. 3. ECQV implicit certificate access elapsed time (a) and memory (b) overheads from CA and from the blockchain

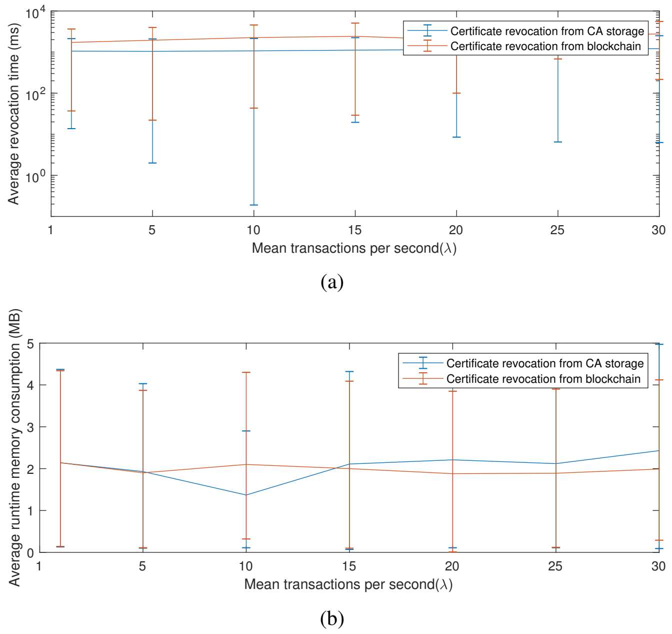


Fig. 4. ECQV implicit certificate revocation elapsed time (a) and memory (b) overheads from CA and from the blockchain

The threat acknowledgement requests triggered aligning to the Poisson arrival using multi-threaded applications. The threat scoring also reflects a limitation on scaling due to the deployment model of blockchain. However a scalable blockchain deployment model can eliminate the limitation significantly.

VI. Implementation and Experimental Results

A. Experimental environment

The computing infrastructure for the experiments consists of Raspberry Pi 3 Model B V1.2 devices connected to the Wireless Local Area Network (WLAN) simulating the 5G

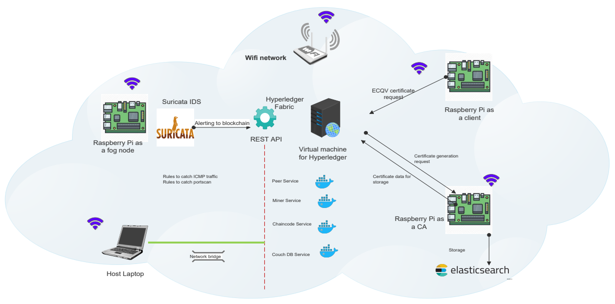


Fig. 5. The experimental setup

network. The Raspberry Pi acts as fog node and the virtual machine as the blockchain service running on the edge computing node. The network is simulated using the Wireless Local Area Network (WLAN). The Raspberry Pi devices connected to the WLAN and the blockchain service are running a virtual machine connected by bridging through the wireless adapter of the host machine. The role of the virtual machine is expected to be identical to the edge computing node running in the network. Figure 5 portraits the experimental setup.

B. Implementation and experimental results

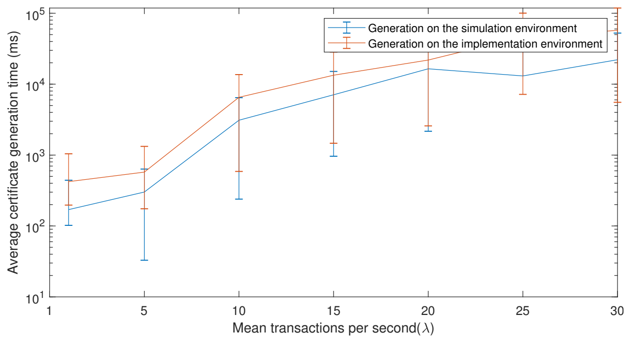


Fig. 6. Comparison of performance of the elapsed time for ECQV implicit certificate generation on the simulation and implementation environment

Figure 6 reflects a comparison between the certificate generation time in the simulation and implementation environment. The averages of the certificate generation time for each transaction rate is higher than the simulation environment. The difference between computational hardware and the latency on the network are the key reasons for the observation.

Figure 7 (a) and (b) illustrate the elapsed time and memory in the implementation environment. The elapsed time averages behave the same as the generation with a minor latency. The memory consumption averages remain lower. The memory efficient techniques of Raspberry Pi implementation is a key reason for this observation.

Figure 8 (a) and (b) illustrate the elapsed time and memory in the implementation environment for certificate revocation. The elapsed time averages behave the same as the generation with a minor latency as per the previous cases. The memory consumption averages remain lower. The memory efficient techniques of Raspberry Pi implementation is the key reason for this observation as per the previous operations.

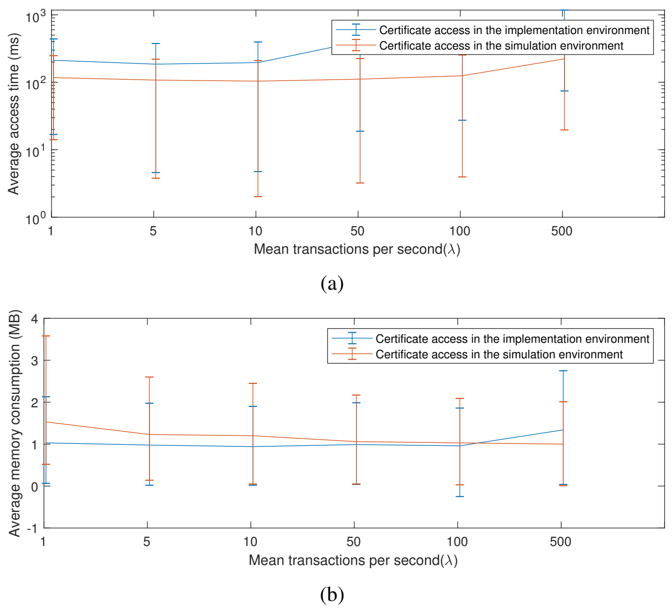


Fig. 7. ECQV implicit certificate access elapsed time (a) and memory (b) overheads in the simulation and the implementation environment

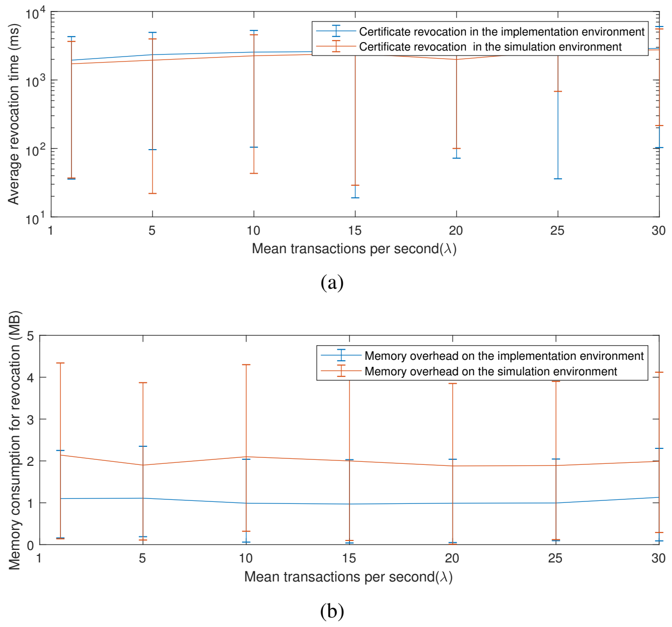


Fig. 8. ECQV implicit certificate revocation elapsed time (a) and memory (b) overheads in the simulation and the implementation environment

C. Limitations of the implementation

The solution deployed in the single order mode in Hyperledger Fabric. The scalability limitations occur due to the deployment mode of Hyperledger. Furthermore, the REST API has some more latency than protocols such as gRPC (Google Remote Procedure Calls) and TCP sockets. Since the limitation

Use Cases	[12]	[13]	[14]	[15]	[16]	Our Solution
Decentralization	No	Yes	Yes	Yes	Yes	Yes
Automated certificate generation	No	Yes	Yes	Yes	Yes	Yes
Edge computing compatibility	Optional	Not specified	Not specified	Not specified	Not specified	Yes
ECQV certificate generation	Optional	No	No	No	No	Yes
Certificate generation financial cost	Service fee	Blockchain fee	Blockchain fee	Blockchain fee	Blockchain fee	No
Threat scoring automatic revocation	No	No	No	No	No	Yes
ECQV certificate generation	Optional	No	No	No	No	Yes

TABLE I
COMPARISON OF CENTRALIZED CA AND FEW EXISTING SOLUTIONS WITH THE PROPOSED SOLUTION

of scaling is entirely dependent on the blockchain platform, it can be eliminated by either deploying more miner nodes or using a plug-and-play scalability improvement technique such as [18].

VII. CONCLUSION

In this paper, we propose a solution to manage the lifecycle of ECQV certificates using the distributed ledger technology. The main advantage is the lightweight nature of ECQV certificate, which enables storage in the ledger. The distributed ledger acts as a tamper proof database and provides faster retrieval of certificates for different purposes. In terms of memory, the lightweight nature of the certificate yields optimal results in related operations. Furthermore, we incorporated smart contracts on the certificate subscription as well as the revocation process. The smart contracts precisely manipulate the threat score and revoke the certificates. The distributed ledger provides transparent records on revocation history for the investigations. The experiment conducted on the Hyperledger Fabric private blockchain platform is not costly to operate. Table I provides a high level comparison between the proposed solution with centralized CA and a few blockchain based PKI implementations.

In order to evaluate the full scalability of the solution, the blockchain platform should be improved. Since the smart contract platform is operating as a separate service, the capability to integrate a user required blockchain platform is ensured. The security improvements of the blockchain, such as scaling up the miners and enforcement of miner security to eliminate 51% attacks can be performed without affecting to the PKI management system. Finally, the feasibility to replace the REST API based architecture should be evaluated by substituting protocols such as gRPC Remote Procedure Call, Constrained Application Protocol (COAP) and Message Queuing Telemetry Transport (MQTT) to eliminate the latency of REST API.

ACKNOWLEDGEMENT

This work is party supported by European Union in RESPONSE 5G (Grant No: 789658) and Academy of Finland in 6Genesis (grant no. 318927) and SecureConnect projects.

REFERENCES

[1] Madhusanka Liyanage, Ijaz Ahmed, Ahmed Bux Abro, Andrei Gurtov, Mika Ylianttila, A comprehensive Guide to 5G Security , Wiley, 2018.

[2] Madhusanka Liyanage, An Braeken, Pardeep Kumar, Mika Ylianttila, IoT Security: Advances in Authentication, Wiley, 2019.

[3] Tharaka Hewa, Anshuman Kalla, Avishek Nag, Mika Ylianttila, Madhusanka Liyanage, ”Blockchain for 5G and IoT: Opportunities and Challenges ”, to be appeared in The 8th IEEE International Conference on Communications and Networking (IEEE ComNet’2020), Hammamet, Tunisia, March 2020.

[4] D.J. Bernstein and T. Lange, SafeCurves: choosing safe curves for elliptic-curve cryptography, https://safecurves.cr.yp.to, accessed 28 March 2019.

[5] D.J. Bernstein, Curve25519: new Diffie-Hellman speed records in public key cryptography, PKC, 9th international conference on theory and practice in public-key cryptography, New York, NY, USA, 2006, Lecture Notes in Computer Science 3958, Springer, pp. 207-228, 2006.

[6] SHA-3 standardization, NIST. Retrieved 2015-04-16.

[7] Certicom Research. 2013. SEC4: Elliptic Curve Qu-Vanstone implicit certificate scheme, Standards for Efficient Cryptography Group. Version 1.0. Retrieved October 30, 2017 from http://www.secg.org/sec4-1.0.pdf, accessed 28 March 2019.

[8] P. Porambage, C. Schmitt, P. Kumar, A. Gurtov, and M. Ylianttila, Two-phase authentication protocol for wireless sensor networks in distributed IoT applications, In IEEE Wireless Communications and Networking Conference (WCNC), Istanbul, pp. 2728-2733, 2014.

[9] D.A. Ha, K.T. Nguyen, J.K. Zao, Efficient authentication of resource-constrained IoT devices based on ECQV 505 implicit certificates and datagram transport layer security protocol, In Proceedings of the Seventh Symposium on Information and Communication Technology, Ho Chi Minh City, 2016, pp. 173-179.

[10] Teniou, A. and Bensaber, B. (2018). Efficient and dynamic elliptic curve qu-vanstone implicit certificates distribution scheme for vehicular cloud networks. Security and Privacy, 1(1), p.e11.

[11] M. Chung and S. Kim, Design of Group Signature-based Payment Protocols for Vehicle Anonymity, Journal of the Korea Institute of Information Security and Cryptology, vol. 29, no. 4, pp. 753773, Aug. 2019.

[12] K. Christidis and M. Devetsikiotis, ”Blockchains and Smart Contracts for the Internet of Things”, IEEE Access, vol. 4, pp. 2292-2303, 2016. Available: 10.1109/access.2016.2566339.

[13] M. Al-Bassam, ”SCPKI: A Smart Contract-based PKI and Identity System”, Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts - BCC ’17, 2017. Available: 10.1145/3055518.3055530

[14] A. Singla and E. Bertino, ”Blockchain-Based PKI Solutions for IoT,” 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC), Philadelphia, PA, 2018, pp. 9-15. doi: 10.1109/CIC.2018.00-45

[15] B. Qin, J. Huang, Q. Wang, X. Luo, B. Liang and W. Shi, ”Cecoin: A decentralized PKI mitigating MitM attacks”, Future Generation Computer Systems, 2017. Available: 10.1016/j.future.2017.08.025 .

[16] A. Yakubov, W. M. Shbair, A. Wallbom, D. Sanda and R. State, ”A blockchain-based PKI management framework,” NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, 2018, pp. 1-6. doi: 10.1109/NOMS.2018.8406325

[17] ”Ethereum/wiki”, GitHub, 2019. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper. [Accessed: 29-Oct- 2019].

[18] Gorenflo, Christian and Lee, Stephen and Golab, Lukasz and Keshav, Srinivasan. (2019). FastFabric: Scaling Hyperledger Fabric to 20,000 Transactions per Second.