Blockchain based Proxy Re-Encryption Scheme for Secure IoT Data Sharing

Ahsan Manzoor*, Madhsanka Liyanage†‖, An Braeken‡, Salil S. Kanhere§, Mika Ylianttila¶
*‡§Centre for Wireless Communications, University of Oulu, Finland
‖School of Computer Science, University College Dublin, Ireland
‡Industrial Engineering INDI, Vrije Universiteit Brussel, Belgium
§School of Computer Science and Engineering, University of New South Wales, Australia
{ahsan.manzoor*, madhusanka.liyanage†, mika.ylianttila¶}@oulu.fi, madhusanka@ucd.ie‖, an.braeken@vub.be‡,
salil.kanhere@unsw.edu.au§

***Abstract*—Data is central to the Internet of Things (IoT) ecosystem. Most of the current IoT systems are using centralized cloud-based data sharing systems. Involvement of such third-party service provider requires also trust from both sensor owner and sensor data user. Moreover, the fees need to be paid for their services. To tackle both the scalability and trust issues and to automatize the payments, this paper presents a blockchain based proxy re-encryption scheme. The system stores the IoT data in a distributed cloud after encryption. To share the collected IoT data, the system establishes runtime dynamic smart contracts between the sensor and the data user without the involvement of a trusted third party. It also uses an efficient proxy re-encryption scheme which allows that the data is only visible by the owner and the person present in the smart contract. The proposed system is implemented in an Ethereum based testbed to analyze the performance and security properties.**

***Index Terms*—Proxy Re-Encryption, Blockchain, Smart Contracts, IoT Data Sharing, Security, Ethereum**

I. Introduction

The Internet of Things (IoT) is an emerging technology which has great technical, social, and economic significance. Current predictions for the impact of IoT are very impressive. With the development of 5G, it is anticipating that 100 billion connected IoT devices will be used by 2025 [1], [2]. It will also have a global economic impact of more than $11 trillion [3] [4].

Data is central to the IoT paradigm. IoT data is collected to serve many different types of applications such as smart home, smart city, wearable, healthcare, smart grid, autonomous vehicles, smart farms, industries and manufacturing, and retail sector [4]–[6]. Therefore, numerous heterogeneous sensors exist to measure a variety of parameters. The collected data from these IoT sensors can be useful for different stakeholders. For instance, air quality measurements are of interest to governmental organizations, application developers and inhabitants of the relevant spaces. However, many challenges arise when organizing this data sharing as these IoT devices, which are typically resource-constrained, require efficient mechanisms to guarantee the data integrity and to enable proper processing and security [7]. Due to the large number of IoT devices, scalable deployment, and maintenance costs [5] should also be taken into account. Currently, almost all the sensor systems upload the data to a centralized cloud and share the sensor data with different stakeholders, who prove access to the cloud storage [8]. The sensors get services from the third-party cloud service provider, such as access control in addition to the data storage. In that case, both sensor and sensor data user have to trust the third-party service provider and also need to pay some fee for their services. In addition, it is needed to establish an agreement between the third-party service provider and sensor data user. Most of these agreements are static and take lots of time and administration to be established [9]. It will result in a significant increase of time before the actual data sharing can be realized [10]. Thus, the current centralized architecture model in IoT systems will struggle to scale up to meet the demands of future IoT systems.

Our Contribution: To solve these issues, we propose a novel blockchain based scheme in combination with a proxy re-encryption mechanism to ensure the confidentiality of the data. Here, the advantage of using blockchain mechanisms to sell the sensor measurements with different users is that the corresponding financial transactions are automatically managed through the agreed smart contract, stored at the blockchain. Moreover, the availability and other quality of service requirements from the legal contract between both parties can be automatically applied. Consequently, compared to the business scenario where the data is stored in a cloud-based infrastructure, there is no need for manual verification of the payments and the predefined requirements. Also, disputes on these aspects are completely avoided.

The remainder of this paper has the following structure. Section II gives an overview of related work. The proposed architecture and proxy re-encryption scheme is explained in Section III and IV. Section V discusses the implementation of the proposed scheme. The performance analysis results are presented in Section VI. Finally, Section VII presents our conclusions.

II. Related work

There exist different studies on the security and privacy of the IoT [11]–[16] and the vast majority of this research work is on understanding and identifying these threats [17]–[21]. Moreover use of blockchain to secure various IoT Platforms were discussed in [22]–[26]. The IoT devices sense, gather

and share a large amount of data, thus opening up significant security and privacy concerns. Khan and Salah [27] in their paper have reviewed different security challenges to IoT and identified insecure transferring of IoT data as a high-level security risk. Authors in [28] demonstrated the lack of basic security by hacking off-the-shelf smart home IoT devices.

In 1998, Blaze, Bleumer, and Strauss [29] initially introduced the concept of proxy re-encryption and constructed the first bidirectional proxy re-encryption application. Authors in [30], [31] also propose a similar scheme but it is not dynamic, hence making it unsuitable for cloud data sharing. In [32], a very efficient solution for data storage in the cloud is proposed using a pairing free proxy re-encryption scheme. However, the scheme is not implemented in practice. Although the underlying structure of our proposed scheme is based on it, some important modification like the inclusion of metadata is included to ensure a practical usage of the scheme.

Most of the prior work partly addresses the problem of securely sharing the IoT data. It is nearly impossible to come up with device-embedded security to solve all the security threats to the IoT devices. Limited computing and power resources of IoT also make the execution of complex security algorithms harder on the device. We propose using the combination of a blockchain and a paring free proxy re-encryption scheme to provide a trading platform and to ensure secure transfer of the sensor data to the user.

III. Proposed Architecture

In this section, we present our new architecture based on the mechanisms of blockchain and re-encryption for secure storing and sharing of the sensor data. We consider four entities in the system: IoT sensors, data requester, cloud provider, and the blockchain, as shown in Figure 1.

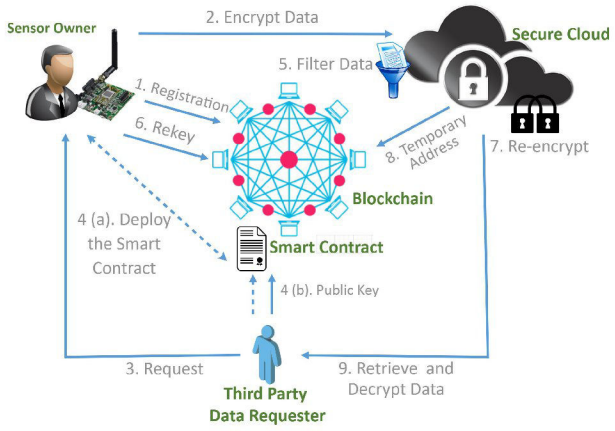

Fig. 1: Proposed Architecture

1) The sensors' owner activates the sensors, and registers them on the blockchain via a smart contract function.
2) After successful registration, the sensors' owner provides the sensor with the required key material such that the measured data can be sent encrypted to the cloud storage server.
3) A user requests access to one (or a group of) sensor(s) of the owner via the smart contract function.
4) After receiving the request, the sensors' owner and requester come to an agreement, a smart contract is generated and mined on the blockchain. The requester interacts with the blockchain to share the public cryptographic key and manages all the financial associated transactions.
5) On receiving the user request, cloud storage is notified by the blockchain. The software then filters the data according to the request.
6) Re-encryption cryptographic key from the sensor owner is updated on the smart contract when the user request is received.
7) Cloud server decrypts and re-encrypts filtered data, before storing it again on a temporary location onto the cloud server.
8) The encrypted data is temporally stored on the server and a transaction containing the address of the stored data is mined on the blockchain.
9) When the data is ready, the requester is notified of the temporary location by the blockchain. The requester can decrypt the data using its private cryptographic key.

IV. Security Aspects

We propose to apply a Certificate Based Proxy Re-Encryption (CB-PRE) scheme, which constitutes of seven polynomial-time algorithms: Setup, CertifiedUserKeyGen, Encrypt, ReKeyGen, ReEncrypt, Decrypt1, and Decrypt2. We now explain each of these phases into more detail. In our proposal, we have combined the phases UserKeyGen and Certify to one phase called the CertifiedUserKeyGen phase.

- SetUp(l): Given a certain security parameter l, the following steps will be executed to derive the public parameters $params$ and the master secret key msk.
  - First, the CA chooses an l-bit prime q. Next, an EC of order q is generated, and a corresponding generator point P is defined. Denote by G the group of EC points.
  - A random value $\alpha \in F_q^*$ is chosen and $P_\alpha = \alpha P$ is computed.
  - Four different hash functions are determined. $H_1 : G \times \{0,1\}^{32} \rightarrow F_q^*$, $H_2 : F_q^* \times \{0,1\}^{64} \rightarrow F_q^*$, $H_3 : \{0,1\}^{64} \times G \rightarrow F_q^*$, $H_4 : F_q^* \times \{0,1\}^{64} \times \rightarrow F_q^*$.
  - The public parameters are now $params = \{G, q, P, P_\alpha, H_1, H_2, H_3, H_4\}$ and the master secret key is put as $msk = \alpha$.
- CertifiedUserKeyGen($params, id_U$): This algorithm is based on the Elliptic Curve Qu Vanstone (ECQV) certificate mechanism [33] and consists of the following three phases:
  - First, the involved entity id_U generates a random value $r_U \in F_q^*$ and computes $R_U = r_U G$. Next the tuple (id_U, R_U) is sent to the CA.
  - Upon arrival, the CA checks the identity of id_U. Next, it also chooses a random value $r_t \in F_q^*$ and computes $R_t = r_t P$. Then the certificate $Cert_U = R_U + R_t$

is derived. Finally, auxiliary information to derive the private key for the involved entity is computed by $r_a = H_1(Cert_U \| id_U) r_t + \alpha$. The tuple $(r_a, Cert_U)$ is sent back.

  - The involved entity computes first its private key $d_U = H_1(Cert_U \| id_U) r_U + r_a$. Its public key equals to $P_U = d_U P$. If $P_U = H_1(Cert_U \| id_U) Cert_U + P_\alpha$, it accepts the key pair (d_U, P_U).

- Encrypt($params, M, id_A, d_A, T_0$): The metadata is generated for the message M, ie. $meta = (id_A \| T_0)$. Next, the following computations are made.

$$\begin{aligned} r &= H_2(d_A \| meta), R = rP \\ C_A &= M \oplus H_3(meta \| rP_A) \\ h_A &= H_4(C_A \| meta) \\ s_A &= r - h_A d_A \end{aligned}$$

  The output C of this algorithm equals to $C = (C_A, meta, h_A, s_A)$.
- ReKey($params, d_A, id_B, Cert_B, C_A, meta$): First $r = H_2(d_A \| meta)$ is derived from C. Then, the public key of id_B is computed as $P_B = H_1(Cert_B \| id_B) Cert_B + P_\alpha$. This leads to the definition of the ReKey as

$$rk_{AB} = H_3(meta \| rP_A) \oplus H_3(meta \| rP_B)$$

  The output is the key rk_{AB}.
- ReEncrypt($params, C_A, rk_{AB}$): The re-encryption phase changes the ciphertext C_A to C_B by

$$C_B = rk_{AB} \oplus C_A$$

  Note that C_B also corresponds to $M \oplus H_3(meta \| rP_B)$, which will be used in the decrypting phase of the delegate. The output C' is now the tuple, containing $C_B, meta, ID_B, h_A, s_A$.
- Decrypt1($params, C, d_A$): Here the delegator wants to decrypt the ciphertext to derive the original message and to check its authenticity. Therefore, the following computations are required:

$$\begin{aligned} r &= H_2(d_A \| meta) \\ M &= C_A \oplus H_3(meta \| rP_A) \\ h_A &= H_4(C_A \| meta) \\ \text{Check: } s_A &= r - h_A d_A \end{aligned}$$

- Decrypt2($params, C', d_B$): In this phase, the delegate B derives the message M from C' by the following operations.

$$\begin{aligned} R &= s_A P + h_A P_A \\ M &= C_B \oplus H_3(meta \| d_B R) \\ \text{Check: } h_A &= H_4(C_A \| meta) \end{aligned}$$

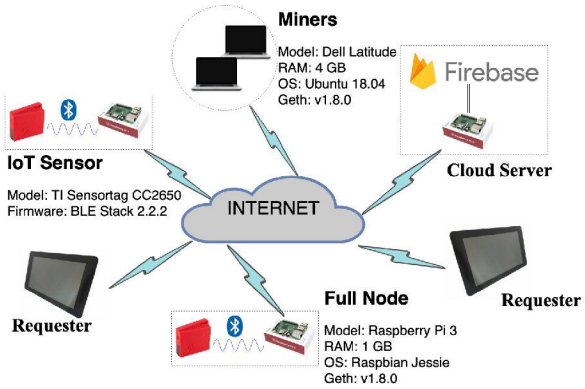

Fig. 2: Overview of the Architecture Implemented

V. Implementation

We demonstrate the feasibility of the system design with the prototype implementation containing a permissioned Ethereum blockchain, IoT sensors and a cloud server for storage of the data. Figure 2 illustrates the setup of the system with three IoT sensors, three mining computers, five ethereum full nodes, two regular users and one cloud storage server.

We configured and connected all the devices to the internet. We used the auto-discovery protocol of Geth to connect the miners and the full nodes, and configured google firebase cloud for storage.

A. Miners

The proposed system consists of three miners that generate a block of transactions on average every 13 seconds. These miners are running on a virtual machine with the same hardware capabilities. All the mining devices were configured to use one Ethereum wallet that collects the mining reward. These miners are running on Geth v1.80 [34] with four mining threads each.

B. Smart Contracts

We developed two smart contracts[1] on truffle [35] and compiled them with Solidity 0.4.24 [36]. The first smart contract consists of the functions to register the sensor, request data, and financial functions. The second smart contract is dynamically created in the runtime when the user requests for the data.

C. IoT Sensors

Each sensor TI Sensortag CC2650 connects to a Raspberry Pi 3 Model B (RSP) through Bluetooth Low Energy, as shown in Figure 2. This RSP manages the sensor and the Ethereum account to perform transactions on the blockchain on behalf of sensors. A sensor application is developed in Python 2.7.12 that connects to the sensor, performs the cryptography functions described in the proxy re-encryption scheme on the sensor data and uploads that data to the cloud storage server. This application synchronizes with the blockchain using the Python- JSON-RPC (JavaScript Object Notation - Remote procedure Calls) library. The MAC address of each sensor acts

[1] https://github.com/ahsan100/smart-contract

as its identity and is used for re-encryption. Once registered, the sensor starts uploading the encrypted data to the cloud server. It is assumed that the BLE connection between sensor and RSP is completely secure.

D. User Application

A customized application is designed as the user interface in Python 2.7.12, running on a Raspberry Pi 3 attached to a touchscreen. This application uses JSON-RPC to get the sensors' information from the blockchain. After selecting the required sensor, the user enters details for specifying the data requirements. We deploy a new smart contract on the blockchain in run-time based on the user-selected options for the requested data (e.g. Sensor selection, Price). This application keeps track of the Ethereum wallet along with ECC [37] secret key of the data requester. The application downloads the data from the cloud server, checks for the signature and integrity, and decrypts the requested data.

E. Cloud Storage Server

The cloud storage server consists of the RSP and the Google Firebase. RSP acts as ethereum full node and connects to the blockchain, while Google Firebase is used for the storage of the data. The authentication and integrity of the data are performed on the RSP and encrypted sensor data along with the meta-data is upload to the Google Firebase in JSON format. This cloud also performs proxy re-encryption and updates the smart contract variable for data address sharing.

VI. Performance Analysis

In this section, we describe the experiments to evaluate the proof of concept implementation. Experiments were designed to study the performance of the framework. We have performed multiple experiments to test the impact of proxy re-encryption on the overall system and performed some scalability tests.

A. Impact of Proxy Re-Encryption

In the first experiment, we measure the impact of proxy re-encryption on the proposed system. The sensor encrypts the data before uploading it to the cloud storage and later re-encrypts it for sharing the data.

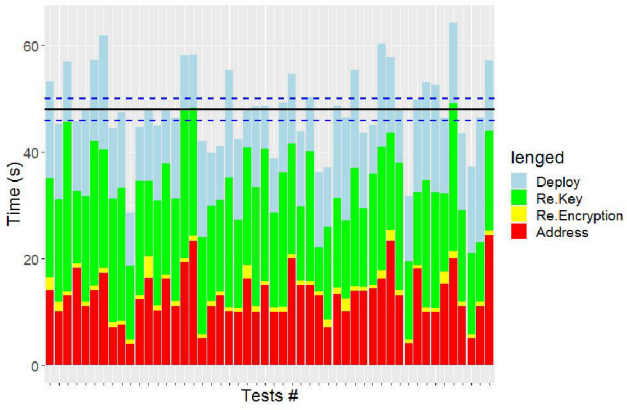

Fig. 3: Impact of Proxy Re-Encryption

In Figure 3, the time of the different parts in the scheme is illustrated. As can be seen, it takes on average 48.01 s to share the encrypted data with the user after the initial request with a confidence interval of 2.07 s. Consequently, adding proxy re-encryption to the scheme increases the delay by 60% due to the mining of the re-encryption key.

B. Scalability

In the second experiment, we measure the scalability of the architecture by performing multiple transactions from multiple requesters to the sensor. The whole process was repeated 10 times for each scenario before taking the average. In the first scenario, only 1 request was initiated by the user and time was measured from the request to the retrieval of data by the requester. In the latter scenarios, the process was repeated by increasing five requests until the overall request reached 50.

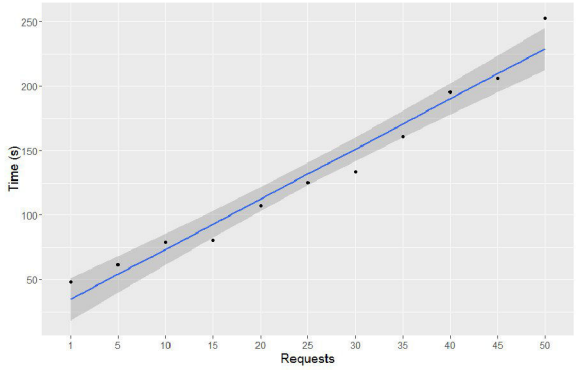

Fig. 4: Scalability Test

As seen from the Figure 4, the process shows a gradual increase in the delay due to the increase of transactions. This increase in the delay is caused by the scalability problem of the Ethereum blockchain. There seems to be a tradeoff between speed and reward for the generation of the new block. The number of transactions mined in a single block of Ethereum blockchain depends on multiple factors such as gas price and limit.

VII. Conclusions

In this paper, we have proposed a blockchain based trading platform with the combination of a paring free proxy re-encryption scheme to ensure secure transfer of the sensor data to the user. We have also validated the proof of concept model on a private Ethereum testbed and demonstrated the practicality of the system design using off-the-shelf laptops and raspberry pis.

In the future, we plan to extend the proposed system with an implementation on a different blockchain platform e.g. Hyperledger. We also plan to extend our architecture by adding a distributed cloud storage to make the system more scalable.

Acknowledgement

This work has been performed under the framework of SECUREConnect, 6Genesis Flagship (grant 318927), RESPONSE 5G (Grant No: 789658), CA15127 (RECODIS) and CA16226 (SHELD-ON) Projects.

REFERENCES

[1] R. Taylor, D. Baron, and D. Schmidt, "The world in 2025-predictions for the next ten years," in *Microsystems, Packaging, Assembly and Circuits Technology Conference (IMPACT), 2015 10th International*. IEEE, 2015, pp. 192–195.

[2] M. Liyanage, I. Ahmad, A. B. Abro, A. Gurtov, and M. Ylianttila, *A Comprehensive Guide to 5G Security*. John Wiley & Sons, 2018.

[3] N. Suryadevara and S. Mukhopadhyay, "Internet of things: A review and future perspective," *Reliance*, 2018.

[4] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, "Survey on multi-access edge computing for internet of things realization," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2961–2991, 2018.

[5] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[6] A. Rajakaruna, P. Porambage, A. Manzoor, M. Liyanage, A. Gurtov, and M. Ylianttila, "Enabling end-to-end secure connectivity for low-power iot devices with uavs," in *in 2nd Workshop on Intelligent Computing and Caching at the Network Edge at IEEE Wireless Communications and Networking Conference (WCNC)*, 2019.

[7] A. Braeken, M. Liyanage, and A. D. Jurcut, "Anonymous lightweight proxy based key agreement for iot (alpka)," *Wireless Personal Communications*, pp. 1–20, 2019.

[8] L. Hou, S. Zhao, X. Xiong, K. Zheng, P. Chatzimisios, M. S. Hossain, and W. Xiang, "Internet of things cloud: architecture and implementation," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 32–39, 2016.

[9] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[10] E. Karafili and E. C. Lupu, "Enabling data sharing in contextual environments: Policy representation and analysis," in *Proceedings of the 22nd ACM on Symposium on Access Control Models and Technologies*. ACM, 2017, pp. 231–238.

[11] J. S. Kumar and D. R. Patel, "A survey on internet of things: Security and privacy issues," *International Journal of Computer Applications*, vol. 90, no. 11, 2014.

[12] K. Zhao and L. Ge, "A survey on the internet of things security," in *2013 Ninth international conference on computational intelligence and security*. IEEE, 2013, pp. 663–667.

[13] M. Ammar, G. Russello, and B. Crispo, "Internet of things: A survey on the security of iot frameworks," *Journal of Information Security and Applications*, vol. 38, pp. 8–27, 2018.

[14] S. Li, L. Da Xu, and S. Zhao, "The internet of things: a survey," *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, 2015.

[15] J. Granjal, E. Monteiro, and J. S. Silva, "Security for the internet of things: a survey of existing protocols and open research issues," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1294–1312, 2015.

[16] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017.

[17] H. Suo, J. Wan, C. Zou, and J. Liu, "Security in the internet of things: a review," in *Computer Science and Electronics Engineering (ICCSEE), 2012 international conference on*, vol. 3. IEEE, 2012, pp. 648–651.

[22] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for iot security and privacy: The case study of a smart home," in *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 2017, pp. 618–623.

[18] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, "A survey on security and privacy issues in internet-of-things," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1250–1258, 2017.

[19] T. Kumar, P. Porambage, I. Ahmad, M. Liyanage, E. Harjula, and M. Ylianttila, "Securing gadget-free digital services," *Computer*, vol. 51, no. 11, pp. 66–77, 2018.

[20] G. Yang, G. Geng, J. Du, Z. Liu, and H. Han, "Security threats and measures for the internet of things," *Journal of Tsinghua University Science and Technology*, vol. 51, no. 10, pp. 1335–1340, 2011.

[21] Y. H. Hwang, "Iot security & privacy: threats and challenges," in *Proceedings of the 1st ACM Workshop on IoT Privacy, Trust, and Security*. ACM, 2015, pp. 1–1.

[23] M. Singh, A. Singh, and S. Kim, "Blockchain: A game changer for securing iot data," in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*. IEEE, 2018, pp. 51–55.

[24] A. Ouaddah, H. Mousannif, A. A. Elkalam, and A. A. Ouahman, "Access control in the internet of things: Big challenges and new opportunities," *Computer Networks*, vol. 112, pp. 237–262, 2017.

[25] Y. Qian, Y. Jiang, J. Chen, Y. Zhang, J. Song, M. Zhou, and M. Pustišek, "Towards decentralized iot security enhancement: A blockchain approach," *Computers & Electrical Engineering*, vol. 72, pp. 266–273, 2018.

[26] R. Ashok, M. Zinopoulou, H. Atlam, G. Wills, N. Zulkipli *et al.*, "Building on a secure foundation for the internet of things," 2016.

[27] M. A. Khan and K. Salah, "Iot security: Review, blockchain solutions, and open challenges," *Future Generation Computer Systems*, vol. 82, pp. 395–411, 2018.

[28] S. Notra, M. Siddiqi, H. H. Gharakheili, V. Sivaraman, and R. Boreli, "An experimental study of security and privacy risks with emerging household appliances," in *Communications and Network Security (CNS), 2014 IEEE Conference on*. IEEE, 2014, pp. 79–84.

[29] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1998, pp. 127–144.

[30] C.-K. Chu, J. Weng, S. S. Chow, J. Zhou, and R. H. Deng, "Conditional proxy broadcast re-encryption," in *Australasian Conference on Information Security and Privacy*. Springer, 2009, pp. 327–342.

[31] M. Sun, C. Ge, L. Fang, and J. Wang, "A proxy broadcast re-encryption for cloud data sharing," *Multimedia Tools and Applications*, vol. 77, no. 9, pp. 10 455–10 469, 2018.

[32] P. Shabisha, A. Braeken, A. Touhafi, and K. Steenhaut, "Elliptic curve qu-vanstone based signcryption schemes with proxy re-encryption for secure cloud data storage," in *International Conference of Cloud Computing Technologies and Applications*. Springer, 2017, pp. 1–18.

[33] M. Campagna, "Sec 4: Elliptic curve qu-vanstone implicit certificate scheme (ecqv)," *institution content-type= institution> Certicom Res</institution>., Mississauga, ON, Canada, Tech. Rep*, 2013.

[34] "Official go implementation of the ethereum protocol," Accessed: 27.09.2018, uRL: https://github.com/ethereum/go-ethereum.

[35] "Truffle," Accessed: 27.09.2018, uRL: https://truffleframework.com/docs/truffle/overview.

[36] "Solidity, the contract-oriented programming language," Accessed: 27.09.2018, uRL: https://github.com/ethereum/solidity.

[37] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of computation*, vol. 48, no. 177, pp. 203–209, 1987.