On-Off Attack on a Blockchain-based IoT System

Fereidoun Moradi*, Ali Sedaghatbaf, Sara Abbaspour Asadollah, Aida Čaušević, Marjan Sirjani,
School of Innovation, Design and Engineering, Mälardalen University,
Västerås, Sweden
{fereidoun.moradi, ali.sedaghatbaf, sara.abbaspour, aida.causevic, marjan.sirjani}@mdh.se

***Abstract*—There is a growing interest in using the Blockchain for resolving IoT security and trustworthiness issues existing in today's complex systems. Blockchain concerns trust in peer to peer networks by providing a distributed tamper-resistant ledger. However, the combination of these two emerging technologies might create new problems and vulnerabilities that attackers might abuse.**

In this paper, we aim to investigate the trust mechanism of Lightweight Scalable BlockChain (LSB), that is a Blockchain specifically designed for Internet of Things networks, to show that a malicious participant in a Blockchain architecture have possibility to pursue an On-Off attack and downgrade the integrity of the distributed ledger. We choose a remote software update process as an instance to represent this violation. Finally, using the actor-based language Rebeca, we provide a model of a system under attack and verify the described attack scenario.

***Index Terms*—Blockchain, Distributed Trust, On-Off Attack, IoT, Security.**

I. Introduction

The concept of Internet of Things (IoT) has enabled an interaction of various physical devices (i.e., sensors, actuators, mobile phones, etc.) over Internet in order to reach a common goal such as increased efficiency, scalability, user flexibility and satisfaction, while potentially decreasing the cost of system development and maintenance. It is expected that in the near future, over 50 billion devices will be connected to the Internet, supporting several different applications like smart cities, smart houses, autonomous cars, precision agriculture, etc., [1]. However, given the heterogeneity, complexity, and possibility to be exposed to different threats, security becomes one of the key concerns in such systems [2].

Blockchain is a cryptographic-based distributed ledger technology that utilises peer to peer (P2P) network communication to enable trusted communication between untrusted communication participants [3]. All transactions within a Blockchain are recorded as a chain of blocks and managed by all network participants without central authority using a distributed cryptography protocol. In such communication, all participants need to validate the information to be appended to the Blockchain. Over the recent years [4], [5], Blockchain technology has been the focus of researches due to its attractive features including trust and security characteristics, decentralisation, anonymity, etc., making it appealing for use in different domains such as banking, smart contracts, smart cities, as well as cybersecurity, especially in the context of IoT [6]–[9].

Although, it can be seen as a promising approach to improve security, information protection, and reliability of IoT devices, it also introduces a new set of challenges such as a choice of an appropriate consensus mechanisms in Blockchain to be applied. The consensus mechanism utilized in Blockchain makes the new generated block difficult to tamper through the hash of previous blocks. A consensus mechanism contains rules and verification procedures to validate provided data and enable participants in the network to put data into the Blockchain. In regards to Blockchain requirements, various consensus mechanisms such as Proof-of-Work (PoW) or Proof-of-Stake (PoS) exist. PoW requires each participant node to provide the proof that the work done and submitted by it qualifies the node to get the right to add new transactions to the Blockchain. The process is expensive and time-consuming as it relies on a complex mathematical calculations, but once it is done the solution can be easily verified by other participants. On the other hand, PoS involves a process of allocation of responsibility when maintaining the public ledger to each node in proportion to the number of virtual currency tokens held by it [10]. Since a consensus mechanism is the core component in the design of Blockchain, the choice of it makes an impact on the performance, scalability, as well as security.

Blockchain technology itself is computationally expensive due to resource-intensive process related to the choice of the consensus mechanism. Since it does not scale well, it might introduce considerable bandwidth overheads and possible delays that are not acceptable for IoT, therefore a novel *Lightweight Scalable Blockchain (LSB)* framework has been proposed in [11]. In our paper we choose to evaluate the suitability of this approach in context of IoT focusing on exploring potential vulnerabilities. Using *Rebeca*, an actor-based modeling language, we provide a model of LSB algorithm in an example of a *Remote Software Update* and introduce a specific type of an attack, called an *On-Off attack* [12]. Furthermore, we use the supporting model checking tool to verify the property of interest and show the possible violation in LSB.

The paper is organized as follows. We first review some relevant related work in Section II. Next, in Section III we introduce necessary background and define concepts used in this paper. The On-Off attack scenario applied to the LSB in context of Remote Software Update example is described in Section IV. Section V provides details of the Rebeca model as well as results of the property verification. Finally, in Section VI we conclude this paper and discuss potential future directions.

II. Related Work

In this section we provide an overview of the literature with focus on how to utilize modeling approaches to verify the security requirements of different Blockchain technology platforms. We start with the review of some of the related works with respect to the modeling Bitcoin protocol [3] where the assessment approaches are similar to our work.

Beukema et al. [13] present a formalized model of Bitcoin protocol in the language mCRL2 as a transition system between agents. The analyzed model contains the behavior of the network, and shows how the consensus protocol behaves when a message corruption and a double-spending attack occurs. Later, Bastiaan et al. [14] propose a stochastic analysis of the Bitcoin protocol, and provide a model through continuous-time Markov chains. The main contributions of their work are analysis of mining algorithms and on a solution to prevent 51% attack.

Ellervee et al. [15] propose a comprehensive model to describe Blockchain by using properties like actors, roles, services, processes and data model. They build a model to guide the business analysts and help them to communicate with the developers. Their comparison between four Blockchain technology platforms include Bitcoin, MultiChain, Ethereum and Chain Core results in explicit understanding of the technology and thei work processes.

A model checking approach that allows reasoning about networks of complex real-timed systems with a stochastic semantic, called UPPAAL SMC [16] also enables some analysis on Bitcoin protocol. Chaudhary et al. [17] investigate the correctness of the Bitcoin protocol by UPPAAL SMC. They calculate the success probability of a double spending attack that is related to the computational power of the attacker. The work examines the Bitcoin protocol and provides its formalization as an UPPAAL model. They conclude that the probability where a fake transaction is inserted in the longest chain, depends on the number of confirmations. Moreover, Fehnker et al. [18] analyze how long does it take for an attacker to succeed in creating a fork and splitting the main chain in a Blockchain. Their work indicates that twenty percent of network miners hash-rate is enough for an attacker to achieve the goal within a few days.

Further work on verifying smart contracts can be found in [19]–[21]. Kosba et. al. [19] verify privacy properties using formal methods and present a programming language Hawk for writing smart contracts. Hawk is a framework for building privacy preserving smart contracts where a programmer can easily write a Hawk program without having to implement any cryptography. Luu et. al. [20] focus on the language for smart contracts and analyze vulnerabilities. They show different smart contracts stored on the Blockchain are potentially vulnerable and may lead to financial losses. Bhargavan et. al. [21] propose a framework to convert smart contracts in Ethereum to a subset of Solidity code and analyze to find the flaws.

III. Background

In this section we introduce the necessary background needed to understand the remainder of the paper. We start with presenting an overview on Blockchain with focus on Lightweight Scalable Blockchain (LSB) [11], [22] as a Blockchain solution for IoT security. We proceed with a brief description of the Rebeca modeling language, used in this paper to model and verify an example of On-Off attack on LSB.

A. Lightweight Scalable Blockchain (LSB)

Blockchain is a ledger distributed among several nodes organized in a P2P setting. Each node has a copy of the ledger and updates it independently [23]. The ledger content consists of a set of blocks, chained together by a cryptography hash value. Each block records a set of transactions. In order to make sure that the order of transactions is the same in all the ledger instances, a consensus mechanism is used by the network participants. Generally, only few nodes have enough processing and storage capacity to enforce the consensus mechanism. These nodes are called miners [24].

LSB is a Blockchain designed specifically for IoT networks. Since typical Blockchain consensus mechanisms require high computational power that makes them difficult to use in IoT. To tackle this problem, Dorri et al. [11] have proposed a time-based block generation algorithm. In this algorithm, each miner is not allowed to add more than one block to the ledger in a given period of time. Further, to reduce overhead, IoT nodes are clustered, and the cluster heads (CHs) are put responsible for verifying transactions and mining blocks. CHs decrease the computation time needed for verifying transactions by using a distributed trust algorithm.

The proposed algorithm relies on direct and indirect evidences to assess the trustworthiness of miners. These evidences are gathered by studying the previous behavior of each miner. If the previous behavior indicates that a miner can be trusted to some degree, then other miners can skip verifying some of the transactions verified by that miner. The exact number of the skipped transactions is recorded in a table maintained by all miners.

For illustration, let us assume that a CH node *A* verifies a block generated by a CH node *B*. At this moment, the node *A* has a direct evidence about the node *B*, and depending on the verification result, it might increase or decrease its trust on the node *B*. Now, suppose that before verification, the node *A* is informed that some other CH nodes have signed the block as valid. Based on this, the node *A* uses this information as an indirect evidence about the node *B*. Table I shows an example trust table that maps the number of positive evidences to the percentage of transactions that should be verified.

Table I: An example for trust table [22].

# Successfully verified blocks	10	20	30	40	50
#% Transaction should be verified	80%	60%	40%	30%	20%

B. Rebeca Modelling Language

Rebeca [25]–[27] is an actor-based modeling language with formal foundation used for modeling concurrent and reactive systems with asynchronous message passing. A Rebeca model consists of the definition of reactive classes and the instantiation part which is called main. The main part defines instances of reactive classes, called rebecs. The behavior of a rebec is determined by its message servers.

In Rebeca, computation is event-driven with messages that can be seen as events. Each rebec takes a message from its message queue and executes the corresponding message server. Communication takes place by asynchronous message passing, which is non-blocking for both sender and receiver. The behavior of a Rebeca model is defined as the parallel execution of the released messages of the rebecs. Rebeca comes with a formal semantics that makes it suitable for model checking purposes. Additionally, the language uses different abstraction techniques to preserve a behavioral specification in temporal logic, while reducing the state space of the model, and verifying desired properties.

IV. On-Off Attack Scenario

In an On-Off attack, it is assumed that a malicious node first behaves honestly to earn the trust from other nodes, and afterwards launches an attack. Since most of the trust models focus on the current node behavior rather than examining past and recent node activities, they are vulnerable to On-Off attacks [28]. As a result, a malicious node can easily hide the misbehavior history by appearing as an honest node or staying idle for some time periods to increase its trust value. In this section, we elaborate an On-Off attack against the distributed trust algorithm proposed by Dorri et al. [22].

Note that the current behavior of a node is considered, while a node can frequently change its behavior from good behavior to bad behavior and vice versa. Suppose a malicious node behaves in the following three phases.

1) *Good Behaviour*: a malicious node generates new valid blocks normally to earn a high level of trust.
2) *Bad Behaviour*: the trust rate of a malicious node achieves the highest value after the period of honest behavior. Afterwards a malicious node abuses the trust rate to insert a fake transaction while building a new block.
3) *Waiting Period*: the malicious node does nothing if the fake transaction was successfully inserted.

Whenever the bad behavior is detected by other nodes during the waiting period, the malicious node again starts behaving well to gain their trust. The probability of detection can be determined based on the mapping provided in the trust table. Considering the mapping in Table I, let us assume that the malicious node generates 50 blocks in the first phase. Based on this only 20% of the transactions generated by this node will be verified by other nodes.

Due to this vulnerability, few malicious nodes can carry out a massive attack on the system by following the above described scenario periodically until the fake transactions are placed in the distributed ledger.

The commonly used mitigation techniques that address an On-Off attack, introducing a factor which considers a weight for nodes performing actions during the times. In other terms, only weight of measured misbehavior is considered rather than periodicity of the misbehavior. In [29] and [12], the authors define an adaptive forgetting scheme and a penalty policy where these components takes both long-time interaction and consistent good behaviors to build up the trust.

In the following, with a simple example, we show how a malicious node among three honest nodes can disturb the integrity of an IoT system. Here, we take a Remote Software Update (RSU) process as an instance. This process utilizes the security and scalability advantages of LSB to ensure the integrity of the software binary hash stored in a distributed ledger [30]. We assume the following constituent systems to be involved in such a scenario: *smart vehicles*, *Original Equipment Manufacturers (OEMs)*, and a *service provider (SP)*. Together, they form a structure of a distributed ledger over a P2P network, enabling them to communicate with each other.

Among the four nodes shown in Figure 1, the dishonest service provider acts as a malicious peer and runs the On-Off attack scenario. The node changes its behavior when the trust rate reaches the highest value, and broadcasts a generated block with a fake transaction to the network. The fake transaction includes hash of a harmful binary code that already has been maintained in the cloud storage as a freshly updated package.

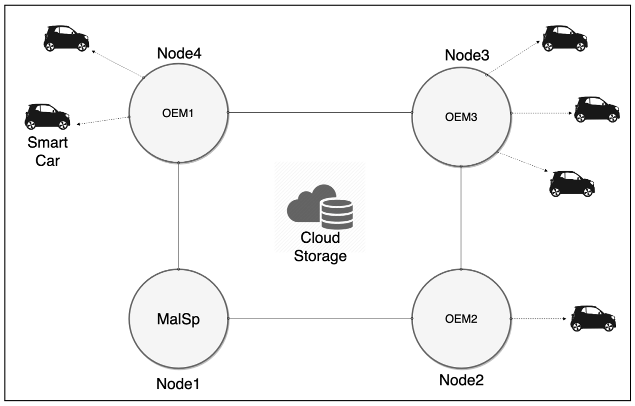

Figure 1: On-Off attack over the RSU process.

The cloud storage is a spot where the Blockchain manages the changes and authenticity of the data, enabling smart cars to directly download the updated software versions. As illustrated in Figure 1, the attack is executed as follows:

1) *Node1* behaves honestly and increase the trust rate.
2) *Node1* uploads a harmful update to the cloud storage.
3) *Node1* inserts the hash value of the harmful code as a transaction into a block and broadcasts the infected block to other nodes.

4) *Node2*, *Node3* and *Node4* receive the block and randomly verify a part of the block. If the nodes do not realize the fraudulent behaviour, the block will be stored.
5) The infected block is stored in the Blockchain, the nodes inform the smart cars about the new update. After that, the smart cars begin downloading the harmful package.

V. Modeling Distributed Ledger Components

In this section, we describe the Rebeca model of the On-Off attack scenario, explained in Section IV. The designed Rebeca model can detect malicious behaviors in LSB. This model includes the behavior of CH nodes involved in the remote software update process.

Listing 1 contains a summarized Rebeca model that consists of two actors: *OEM* and *MalSp*. The actor *OEM* is a CH node of LSB and responsible to build a block and broadcast the block to other *OEMs*. The actor *MalSp* is a CH node of LSB, as well, and builds blocks properly to find the opportunity to insert a fake transaction. Both *OEM* and *MalSp* store transactions in a pending pool, and build a block when the size of the pool is equal to the determined size. In our case we assume to be equal to three.

The model declares instances of the actors with message servers. The collaboration between actors makes possible to store transactions by passing messages to the message servers. We define *buildblk()*, *addtoBC()* and *ack()* message servers for both *OEM* and *MalSp* actors. The *buildblk()* of *OEM* properly follows the consensus and trust algorithm, takes transactions from the pending pool and builds a new block. However, the *buildblk()* of *MalSp* examines the trust rate while building a new block to find opportunity for exploiting the distributed trust algorithm and transmits the infected block to the Blockchain (i.e., a block with at least one fake transaction). We consider the numbers from 0 to 2 for low, high and highest degree of trust rate, respectively. The *MalSp* behaves the same as a *OEM* behaviour until its trust rate becomes high enough. Both *OEM* and *MalSp* have a *addtoBC()* message server to verify transactions in a block. This message server receives broadcast blocks and operates block verification process. If the block verification finishes successfully, the valid block can be stored in the Blockchain. The block generator must be informed of the block verification status through a message. The *ack()* is defined for both *OEM* and *MalSp*. This message server notifies the block generator by sending an acknowledgement message to show the confirmation for appending the block to the Blockchain.

In order to model check the Rebeca model, we need to define the property that we want to verify. The result of a property check tells us whether the property was satisfied or not. We aim at checking whether all the transactions inserted into the Blockchain are valid. In our Rebeca model, a Blockchain is defined as a list of blocks where each block is recorded as an 8-tuple ($MinerID, BlockID$, $GenID1$, $TxID1$, $GenID2$, $TxID2, GenID3, TxID3$), where:

- $MinerID$ specifies the ID of the miner that has generated the block.

```
// OEM (Miner) actor
reactiveclass OEM(150) {
    knownrebecs {
        OEM oem1;
        OEM oem2;
        MalSp m;}
    OEM(int myId) {}
    msgsrv buildblk(int BlockID) {
    // Checks its pending pool to build a new block
        if (txsNum == 3) {
        //Appends to the BC and broadcasts
            oem1.addtoBC(B);
            oem2.addtoBC(B);
            m.addtoBC(B);}
    }
    msgsrv addtoBC(int B) {
    //Checks trust rate and validates trxs }
    msgsrv ack(int BlockID){}
}
-----------------------------------------------------
// Malicious Sp (Miner)
reactiveclass MalSp(150) {
    knownrebecs {
        OEM oem1;
        OEM oem2;
        OEM oem3;}
    MalSp(int myId) {}
    msgsrv buildblk(int BlockID) {
        //Checks its pending pool and trust rate
        //Build an infected block
        if ((txsNum == 2)&&(trustRate == 2)) {
        //Appends to the BC and broadcasts
            oem1.addtoBC(B);
            oem2.addtoBC(B);
            oem3.addtoBC(B);}
    }
    msgsrv addtoBC(int B) {
    //Checks trust rate and validates trxs}
    msgsrv ack(int BlockID){}
}
```

Listing 1: The summarized Rebeca model.

- $BlockID$ is the ID of the block in the Blockchain.
- $GenID1$ is the ID of the first transaction generator.
- $TxID1$ is the ID of the first transaction.
- $GenID2$ is the ID of the second transaction generator.
- $TxID2$ is the ID of the second transaction.
- $GenID3$ is the ID of the third transaction generator.
- $TxID3$ is the ID of the third transaction.

As an instance, in the Rebeca model the block $[3, 4, 2, 3, 2, 4, 2, 5,]$ includes identification numbers of the block, the transactions, and the generators. The CH node with identification number 3 ($[3, *, *, *, *, *, *, *,]$), takes transactions with identification numbers 3 to 5 from the pending pool and builds the block($[*, *, *, 3, *, 4, *, 5,]$). The number 4 shows that this block is located in the second chain of Blockchain ($[*, 4, *, *, *, *, *, *,]$). These transaction are generated by the node with identification number 2 ($[*, *, 2, *, 2, *, 2, *,]$).

Figure 2 shows a diagram depicting one possible way of interaction between actors leading to the attack scenario described in Section IV. According to this diagram, *MalSp* abuses the highest trust rate and calls *buildblk(B)* to build the infected block B (1). The *MalSp* asks *OEM1* to verify

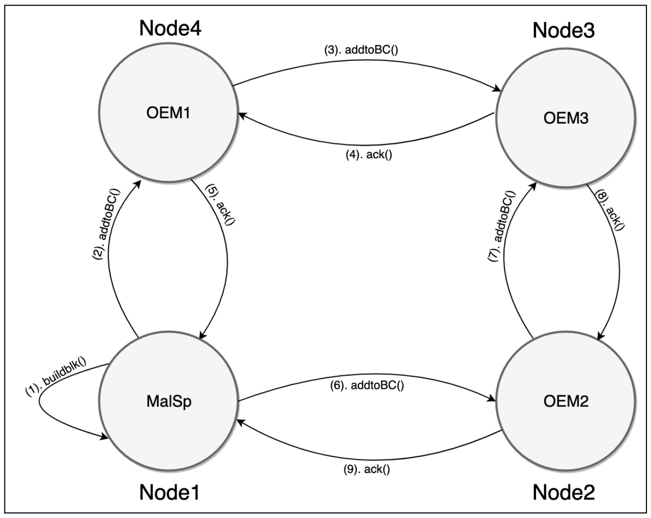

Figure 2: An actor-diagram of the On-Off attack scenario.

the block B and append it to the Blockchain through *addtoBC(B)* message server (2). The *OEM3* is linked to the *MalSp* indirectly and needs to verify the block B. Therefore, the *OEM3* gets the message *addtoBC(B)* from *OEM1* to do the verification (3). Based on the trust rate associated to *MalSp*, *OEM1* and *OEM3* randomly choose a part of the block B for verification and notify *MalSp* by sending *ack(B)* to show the block successfully verified $(4,\ 5)$. The *OEM2* receives *addtoBC(B)* (6) and transfers the message to *OEM3*(7). Both *OEM3* and *OEM2* response back via *ack(B)* $(8,\ 9)$.

Assuming that the trust rate is at the highest value, most of the transactions in the block B would be ignored while the *OEMs* verify the block. In the situation where none of *OEM1*, *OEM2* and *OEM3* have detected the fake transaction, *MalSp* is informed the infected block (block B) appended to the Blockchain by receiving the last *ack(B)*.

The assertion in Listing 2 is defined as a property to check which transactions are ignored in the block verification process. Whenever all *OEMs* ignore the fake transaction, the assertion is failed and indicates this event as a counter-example.

```
Assertion:
!(OEM1.ignoreTx && OEM2.ignoreTx && OEM3.ignoreTx);
```

Listing 2: The property to indicate the fake transaction.

In case where at least one fake transaction is inserted in the Blockchain, it is expected that the Rebeca model checker generates a counter-example indicating the situation in which the described property is violated. Figure 3 shows the chain of blocks with and without fake transaction. Figure 3(a) presents the Blockchain without fake transaction and Figure 3(b) shows a counter-example for the Rebeca model discussed above. The *BC* is an 8-tuple list which is defined for the Blockchain. The Blockchain must store the same valid blocks in each *OEM*. We only present *OEM1.BC* as an example and the *BC* of *OEM2* and *OEM3* are similar to this *BC*. Accordingly, the transaction number 10 in block 5, which is inserted by *MalSp* with number 3 is a fake transaction. Given this, the Rebeca model demonstrates the situation where a vulnerability in LSB leads to a successful On-Off attack. The detials of the designed Rebeca model are uploaded and available at GitHub [31].

$$\begin{array}{l}
\text{OEM1.BC} = [\\
[3,1,2,3,2,4,2,5,],\\
[2,2,2,1,1,1,2,2,],\\
[3,3,2,9,1,2,1,3,],\\
[2,4,2,6,2,7,2,8,],\\
[0,0,0,0,0,0,0,0,],\\
[0,0,0,0,0,0,0,0,],\\
[0,0,0,0,0,0,0,0,],\\
...]
\end{array}$$

(a) Before injecting the fake transaction.

$$\begin{array}{l}
\text{OEM1.BC} = [\\
[3,1,2,3,2,4,2,5,],\\
[2,2,2,1,1,1,2,2,],\\
[3,3,2,9,1,2,1,3,],\\
[2,4,2,6,2,7,2,8,],\\
\boxed{[3,5,3,10,1,4,1,5,],}\\
[0,0,0,0,0,0,0,0,],\\
[0,0,0,0,0,0,0,0,],\\
...]
\end{array}$$

(b) After injecting the fake transaction.

Figure 3: OEM Blockchain before and after injecting a fake transaction.

VI. Conclusion and future work

There is a large number of existing industrial systems that are exploring the possibilities to extend towards the IoT. One of the major concerns in such systems is ensuring security and trust that could be seen as a difficult task, due to resource constrained devices, standards that do not support such shift, or absence of suitable software and hardware solutions applicable in such a setup. Moreover, one has to also account for the dynamic nature of security where changes in the environment and interactions in such complex systems might introduce possibly not expected security-critical situations. One of the ways to mitigate such challenges can be seen in lightweight Blockchain-based solutions for resource constrained IoT devices, such as LSB. LSB focuses on increasing security by using the distributed trust management and at the same time decrease the bandwidth, processing time, and enable services without delays.

In this paper, we study the proposed approach in context of an IoT architecture and example coming from the vehicular domain. We model the approach using the actor based language Rebeca. Our intention with such a model is to explore the suitability of LSB in context of possible attacks, with focus on the On-Off attack. We show how we can use Rebeca as an appropriate testing tool for evaluating security properties in distributed environments and provide a fruitful analysis. It can easily be used to model complex scenarios that involve different agents, as well.

As future work, we plan to model different policies in building the trust over distributed networks and show how the modeling can be useful to check different mitigation techniques to decrease the probability of attacks occurring. Moreover, our aim is to find out the probability of attacks in different configurations when there should be a trade-off between trustworthiness and improving performance. Also, we look forward extending our work towards examining the security terms and features of the infrastructure that is being used by the Blockchain. In that case our focus will shift towards examples of applications that focus on the Internet of Objects and Cyber-Physical Systems.

Acknowledgment

This research is supported by Swedish Foundation for Strategic Research (SSF) via the Serendipity project.

References

[1] L. Atziori, A. Iera, and G. Morabito, "The internet of things: A survey-computer networks," 2010.

[2] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying iot security: An exhaustive survey on iot vulnerabilities and a first empirical look on internet-scale iot exploitations," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2019.

[3] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system, http://bitcoin. org/bitcoin. pdf," 2008.

[4] M. Samaniego and R. Deters, "Blockchain as a service for iot," in *2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 433–436, Dec 2016.

[5] D. Miller, "Blockchain and the internet of things in the industrial sector," *IT Professional*, vol. 20, pp. 15–18, May 2018.

[6] Z. Zheng, S. Xie, H.-N. Dai, and H. Wang, "Blockchain challenges and opportunities: A survey," *Work Pap.–2016*, 2016.

[7] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *Ieee Access*, vol. 4, pp. 2292–2303, 2016.

[8] A. Reyna, C. Martín, J. Chen, E. Soler, and M. Díaz, "On blockchain and its integration with iot. challenges and opportunities," *Future Generation Computer Systems*, vol. 88, pp. 173–190, 2018.

[9] Q. He, N. Guan, M. Lv, and W. Yi, "On the consensus mechanisms of blockchain/dlt for internet of things," in *2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES)*, pp. 1–10, IEEE, 2018.

[10] M. Maroufi, R. Abdolee, and B. M. Tazekand, "On the convergence of blockchain and internet of things (iot) technologies," *arXiv preprint arXiv:1904.01936*, 2019.

[11] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Lsb: A lightweight scalable blockchain for iot security and privacy," *arXiv preprint arXiv:1712.02969*, 2017.

[12] N. Labraoui, M. Gueroui, and L. Sekhri, "On-off attacks mitigation against trust systems in wireless sensor networks," in *IFIP International Conference on Computer Science and its Applications*, pp. 406–415, Springer, 2015.

[13] W. Beukema, "Formalising the bitcoin protocol," in *21th Twente Student Conference on IT*, 2014.

[14] M. Bastiaan, "Preventing the 51%-attack: a stochastic analysis of two phase proof of work in bitcoin," in *Availab le at http://referaat. cs. utwente. nl/conference/22/paper/7473/preventingthe-51-attack-a-stochasticanalysis-oftwo-phase-proof-of-work-in-bitcoin. pdf*, 2015.

[15] A. Ellervee, R. Matulevicius, and N. Mayer, "A comprehensive reference model for blockchain-based distributed ledger technology.," in *ER Forum/Demos*, pp. 306–319, 2017.

[16] A. David, K. G. Larsen, A. Legay, M. Mikučionis, and D. B. Poulsen, "Uppaal smc tutorial," *International Journal on Software Tools for Technology Transfer*, vol. 17, no. 4, pp. 397–415, 2015.

[17] K. Chaudhary, A. Fehnker, J. Van De Pol, and M. Stoelinga, "Modeling and verification of the bitcoin protocol," *arXiv preprint arXiv:1511.04173*, 2015.

[18] A. Fehnker and K. Chaudhary, "Twenty percent and a few days–optimising a bitcoin majority attack," in *NASA Formal Methods Symposium*, pp. 157–163, Springer, 2018.

[19] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *2016 IEEE symposium on security and privacy (SP)*, pp. 839–858, IEEE, 2016.

[20] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 254–269, ACM, 2016.

[21] K. Bhargavan, A. Delignat-Lavaud, C. Fournet, A. Gollamudi, G. Gonthier, N. Kobeissi, N. Kulatova, A. Rastogi, T. Sibut-Pinote, N. Swamy, *et al.*, "Formal verification of smart contracts: Short paper," in *Proceedings of the 2016 ACM Workshop on Programming Languages and Analysis for Security*, pp. 91–96, ACM, 2016.

[22] A. Dorri, S. S. Kanhere, and R. Jurdak, "Towards an optimized blockchain for iot," in *Proceedings of the second international conference on Internet-of-Things design and implementation*, pp. 173–178, ACM, 2017.

[23] M. Crosby, P. Pattanayak, S. Verma, V. Kalyanaraman, *et al.*, "Blockchain technology: Beyond bitcoin," *Applied Innovation*, vol. 2, no. 6-10, p. 71, 2016.

[24] L. Bach, B. Mihaljevic, and M. Zagar, "Comparative analysis of blockchain consensus algorithms," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 1545–1550, IEEE, 2018.

[25] M. Sirjani, A. Movaghar, A. Shali, and F. S. De Boer, "Modeling and verification of reactive systems using rebeca," *Fundamenta Informaticae*, vol. 63, no. 4, pp. 385–410, 2004.

[26] "Rebeca homepage," *http://rebeca-lang.org/Rebeca*, (accessed June 3, 2019).

[27] A. Jafari, J. J. S. Nair, S. Baumgart, and M. Sirjani, "Safe and efficient fleet operation for autonomous machines: an actor-based approach," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, pp. 423–426, ACM, 2018.

[28] H. Alzaid, M. Alfaraj, S. Ries, A. Jøsang, M. Albabtain, and A. Abuhaimed, "Reputation-based trust systems for wireless sensor networks: A comprehensive review," in *IFIP International Conference on Trust Management*, pp. 66–82, Springer, 2013.

[29] Y. L. Sun, Z. Han, W. Yu, and K. R. Liu, "Attacks on trust evaluation in distributed networks," in *2006 40th Annual Conference on Information Sciences and Systems*, pp. 1461–1466, IEEE, 2006.

[30] A. Dorri, M. Steger, S. S. Kanhere, and R. Jurdak, "Blockchain: A distributed solution to automotive security and privacy," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 119–125, 2017.

[31] F. Moradi, "On-off attack actors model," *https://github.com/fereidoun-moradi/On-Off-Attack*, 2019.