Building trust among things in omniscient Internet using Blockchain Technology

Radu BONCEA[1,2], **Ionut PETRE**[1,3], **Victor VEVERA**[1]
[1] National Institute for Research and Development in Informatics - ICI Bucharest
[2] Politehnica University of Bucharest
[3] "Lucian Blaga" University of Sibiu
radu@rotld.ro, ionut.petre@rotld.ro, victor.vevera@ici.ro

Abstract: Since 1999, when Kevin Ashton coined the term Internet of Things (IoT), till our present days, IoT has evolved from a simple concept to one of the topmost business growth drivers. Along with Machine Learning, Cloud Computing and Big Data, IoT is the foundation stone upon which data- driven digital services are built. In near future IoT will reach a tipping point where most of the data generated in Internet will come from billions of devices that are too resource-constrained to be able to efficiently enforce complex security and data privacy policies. The solution is to use lightweight authentication and agreement protocols for dumb devices and integrate distributed ledger technologies, e.g. blockchain in smarter devices and make use of smart contracts to execute processes on predetermined rules. In this regard, we present a simple smart contract written in Solidity and a young, but very promising, blockchain IoT-centric platform, IoTeX. IoTeX is to become fully operational in 2019, it supports Solidity and Ethereum virtual machine and adopts a blockchains in blockchain architecture, with focus on accommodating a large number of transactions than traditional PoW consensus blockchains. And we finish with presenting some interesting tools incubated by Hyperledger, a Linux Foundation project, that allow us to build bottom-top blockchain building blocks.
Keywords: IoT, blockchain, trust model, smart contract, data privacy, data integrity, IoTeX, Hyperledger

INTRODUCTION

In 2018, the number of devices connected to the Internet and used around the world has exceeded 17 billion, of which the number of IoT devices is about 7 billion [1] (excluding smartphones, tablets, laptops, fixed or mobile phones). And the number is expected to grow to 35 billions by year 2025. All these devices produce large quantities of raw data which are stored in data lakes [2], waiting to be processed using Big Data methods such as batch and stream processing, in a Cloud Computing infrastructure in order to build new intelligence-oriented applications and services.

Today's IoT deployment and operating models are largely focused on closed enterprise ecosystems, but as the data grows and becomes the new gold and IoT turns into the new gold mine, there is an increasing interest towards datasharing and adoption of a shared economy model. However, this horizontal model in which raw data are transactioned on digital data marketplaces or interchanged among IoT devices, poses several key challenges in terms

of authorization, authentication, ensuring data privacy and integrity, such as:

1. ***Identification*** is the threat of associating am identifier, such as a name, address or a pseudonym, with a real person and data about that person. The identification is currently most accounted threat during the information processing phase;
2. ***Localization*** and tracking is the threat that relies on the capability of determining and recording a person's location. This threat requires identification of the subject. Most common privacy violations related to this threat include GPS stalking, disclosure of private information, or generally the uneasy feeling of being watched [3, 4];
3. ***Profiling*** is the threat that relies on compiling information dossiers about individuals with the goal of inferring a person's interests and relations with other profiles and data. With the new machine learning capabilities which reduce the costs, mass-profiling is becoming more accessible. Common profile-based privacy violations include price discrimination, social engineering, unsolicited advertisements (e.g. spam campaigns) or erroneous automatic decisions, e.g. by Facebooks automatic detection of sexual offenders [5, 6, 7];
4. ***Privacy-violating*** interaction and presentation is about conveying private information through a public medium and disclosing it to an unwanted audience. This is a common threat in IoT domains such as smart retail, transportation and health care, where there is a need for heavy interaction with the user;
5. ***Lifecycle transitions*** is a threat that occurs when smart things disclose private information during changes in control spheres in their lifecycle. A good example is the compromising photos and videos that are often found on used cameras or smart phones;
6. ***Inventory attack*** refers to the unauthorized collection of information about the existence and characteristics of personal assets, e.g., burglars can use inventory data to check the property to find a safe time to break in;
7. ***Linkage consists*** in linking different previously separated systems such that the combination of data sources reveals (truthful or erroneous) information that the subject did not disclose to the previously isolated sources and, most importantly did not want to reveal.

	Technology	Size	Inter - connection	Data collection	Thing Interaction	System Interaction	Lifecycle	Vertical vs. horizontal
Identification	Cameras, face recognition		Fingerprinting			Speech, cloud interfaces		
Tracking	Indoor LBS			Decreasing awareness		Data trails		
Profiling		Explosion of data sources		Qualitatively new sets of data				
Interaction & Presentation					Presentation media	Pervasive interaction with users		
Inventory attacks	Diversification		Wireless communication					
Linkage				Decreasing transparency				Drives linkage locally

***Table 1**: Summary of impact of the evolving features on the seven categories of privacy threats [8]*

One way to address the security related issues is by implementing standard authorization / authentication models, including two factor authentication (2FA) [9, 10], but there are drawbacks. Storing, rolling out and exchanging tokens and public keys within M2M space should scale, in this case, exponentially, which is far from desirable. If we use a central model, such as an IoT platform, which would allow us to scale linearly, there is still the SPOF (single point of failure) issue. If the platform gets compromised, the whole system is at risk, as already proven by recent data breaches

involving Facebook [11], Google [12], Quora [13], Marriott Hotels [14], just to name a few.

Currently there are two complementary strategies in making future IoT reliable and secure:

1. A strategy that focuses on resource-constrained devices such as IoT sensors, actuators, beacons, devices found in the edge layer (see Figure 1) and which are part of Internet-Integrated Wireless Sensor Networks. Such a solution would be to develop lightweight authentication and key agreement protocols at gateway or edge layers [15];

2. The second method is focused on gateways and cloud platforms, where data has been filtered, aggregated, pruned and stored in specific formats. These layers have more resources, including more processing power and can encapsulate the business logic of sharing data. The solution is to use a decentralized model such as blockchain and the power of smart contracts.

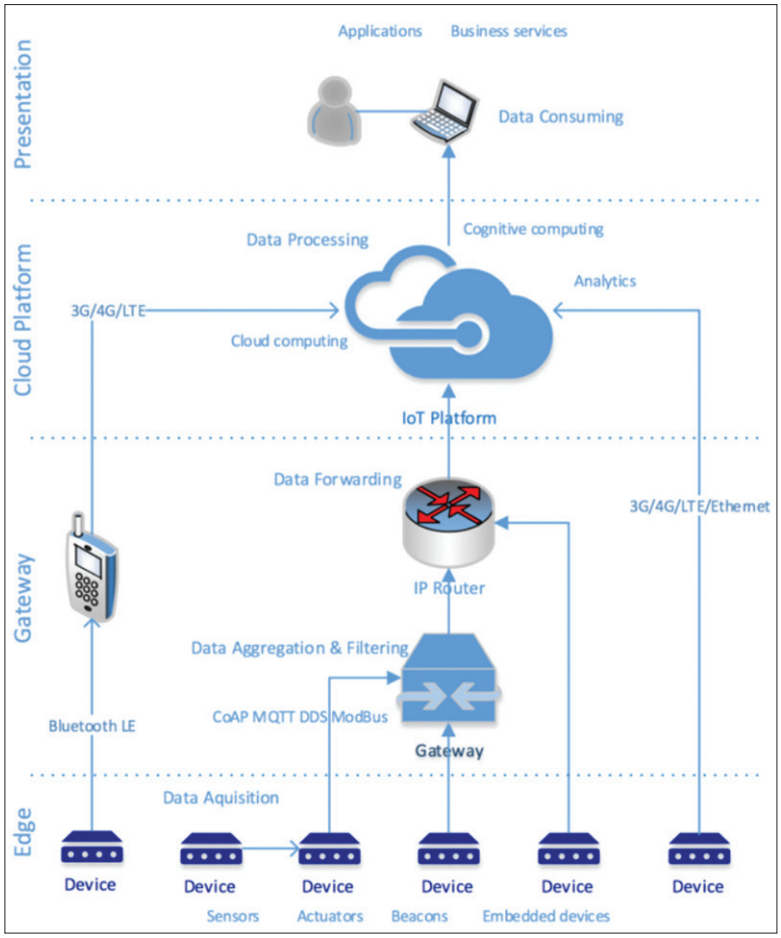

***Figure 1**: Holistic view of IoT generic architecture [16]*

BUILDING BLOCKS

BLOCKCHAIN

Blockchain is an open, distributed, single, shared, tamper-evident ledger for maintaining permanent records of transactional data. The records are called blocks and are linked using cryptography. Blockchain is a linked list, where the new block contains a cryptographic hash (link) of the previous block, and the timestamp and transaction data. It is pinning the following principles [17]:

1. ***Distributed Database***. Each party has access to the ledger and keeps a full copy of the database. No single participant controls the information or data. This makes it possible for each participant to validate the records of its transaction partners directly, without any third-party partner;

2. ***Peer-to-Peer Transmission***. Blockchain is managed by a peer-to-peer (P2P) network collectively adhering to a protocol for inter-node communication. Nodes in P2P network validate transactions (adding new blocks) by consensus, following economic incentives such as proof of work consensus algorithm;

3. ***Transparency***. Each transaction and its associated data are visible to anyone with access to the system. Each node and each user have a unique 30-plus-character alphanumeric address, which is used for identification. Users can opt for anonymity or provide proof of their identity to others. Transactions occur between blockchain addresses;

4. ***Irreversibility of Records***. Once a transaction has been saved in the ledger, the records cannot be tampered. As a result, they are synced to each transaction record that was posted in the past. Various machine algorithms and approaches are enforced to ensure that the storage of information

is permanent, chronologically ordered, and readily available to any or all others on the network;

5. ***Computational Logic.*** Blockchain transactions can be tied to computational logic and in essence programmed. Therefore, users can set up algorithms and rules that automatically trigger transactions between nodes.

CONSENSUS ALGORITHMS

In block, the validation of transactions is done by the P2P network of computers using a consensus protocol instead of relying on a single trusted third party. The blockchain protocol formalizes pre-defined consensus rules for validating transactions on the P2P network, as hard-coded governance rules, managing and auto-enforcing transactions of all participants in the network. Some of the most important consensus algorithms are:

1. ***Proof of Work (PoW)*** [18] was first used by Bitcoin and is known as the mining process. Miners would have to solve complex mathematical puzzles through trial and error, a process that requires a lot of computational power, and the first miner that solves the puzzle would be given the permission to add the transaction block to blockchain. And he would be rewarded for this expensive operation. Thus, to have access to block creation and transaction validation, an attacker would need at least half of the processing power in the P2P network. In the case of Bitcoin, the attacker would have been able to calculate 24 petahashes and pay $150 for electricity every second;

2. ***Proof of Stake (PoS)*** [19] has a different approach than PoW. Block creators are chosen randomly from a pool of stakers (users that stake their tokens to become a validator, locking their wealth for a certain time). That means all tokens/coins are generated at the very beginning, instead of being mined like in PoW. The validators or block creators are also rewarded proportionally to their stake. PoS is not only more energy efficient than PoW, it also has another major distinction. In PoW it is possible for miner to not own the coins they are mining, meaning they only seek to maximize their profits without actually improving the network. In PoS, block creators have their share of interest to maintain the network as they actually hold the coins of the blockchain on which they are validating. The PoS algorithm scales better and provides higher transaction throughput, making it more IoT friendly. The downside of PoS is the compromise on a lesser security than PoW. An attacker could in theory buy enough stakes to become the majority and thus validate wrong transactions as part of the attack. However, this scenario is unlikely, due to economic constraints. Buying so many stakes would certainly result in hyperinflation to the point the cost of the attack outweighs the reward. Recently, Ethereum announced the transition from PoW to PoS and a new protocol (Casper) that handles the reward and punishment, by seizing the stakes of malicious validators;

3. ***Delegated Proof-of-Stake (DPoS)*** [20] is known as the democratic blockchain. In DPoS, users can stake their tokens to vote for certain delegates. The vote weight is proportional with the user's number of coins (e.g. if A stakes 2 coins for a delegate and B stakes 1 coin, A's vote outweighs B by 2 times). The delegate with most votes is allowed to create new blocks and to receive the reward, that could be a fix amount generated through inflation or based on transaction fee. Therefore, a delegate wants to receive as much votes as possible, constantly seeking to create things valuable for the network and the community. One use case is EOS, the crypto currency powering the EOSIO, an open-source blockchain software protocol that provides developers and entrepreneurs with a platform on which to build, deploy and run high-performing decentralized applications (dApps);

4. ***Practical Byzantine Fault Tolerance (pBFT)*** [21] model provides with a Byzantine state machine replication that tolerates Byzantine faults (e.g. compromised nodes) based on the assumption that there are nodes that have failed and there are manipulated messages delivered in the network with the purpose to create confusion. The pBFT algorithm has been designed to work asynchronously and has been optimized for high performance with an impressive overhead

runtime and only a slight increase in latency. All the nodes in the system in pBFT model are ordered in a sequence with one node being the primary or the leader node and the others referred to as the backup nodes. All nodes within blockchain exchange messages with each other in order for honest nodes to come to an agreement of the state of the system through a majority. Nodes communicate with each other heavily, and not only have to prove that messages came from a specific peer node, but also need to verify that the message was not modified during transmission [22]. pBFT is more efficient than PoW, but the model only works well with small consensus group sizes due to the cumbersome amount of communication that is required between the nodes. pBFT is optimal for smaller blockchains. There are several platforms that have implemented pBFT such as Linux Foundation Hyperledger Fabric.

SMART CONTRACTS

Smart contracts are lines of code or small programs that are stored on a blockchain like any other transaction and automatically execute when predetermined terms and conditions are met. Basically, smart contracts work by implementing simple "if/when... then... " statements that are written into code on a blockchain in a specific programming language. One such language is Solidity [23], a high level and object-oriented language designed to target the Ethereum Virtual Machine (EVM). In Figure 2 we demonstrate a simple contract between an IoT service provider and potential service consumers. A user that wants to access the service, has to enroll and deposit a minimum of 10 Ethereum coins (the enroll method of the contract). The user will then use the getaccess method to reserve the right to use the service for a certain number of hours. The contract will calculate a fee which will be subtracted from user's balance and transfer to service owner wallet. Users could also deposit additional coins using deposit method, check their balance using balance method or withdraw coins from their balance. Please notice the depositBalance method, which can be invoked only by the service owner to check the total balance of all users. All actors involved in smart contract transactions need blockchain accounts (blockchain wallets), each account being identified by a unique address (public key). Running each contract requires ether transaction fees, which depend on the amount of computational.

Use cases for smart contracts include "multi-signature" accounts, where funds are spent only when a required percentage of people agree. They can also be used to manage agreements between users, say, if one buys insurance from the other or access certain services. Smart contracts could be used by other contracts similar to how a software library works or could be used as an application state repository, to store information about an application.

```
pragma solidity ^0.4.22;

contract SimpleIoTDataService {
    uint8 private clientCount;
    uint private hourly_fee;
    mapping (address => uint) private accounts;
    address public owner;

    event LogDepositMade(address indexed accountAddress, uint amount);

    constructor() public {
        owner = msg.sender;
        clientCount = 0;
        hourly_fee = 1 finney;
    }

    function enroll() public payable returns (uint) {
        require(msg.value >= 10 ether, "minimum 10 ether service access");
        clientCount++;
        accounts[msg.sender] = msg.value;
        return accounts[msg.sender];
    }

    function deposit() public payable returns (uint) {
        accounts[msg.sender] += msg.value;
        emit LogDepositMade(msg.sender, msg.value);
        return accounts[msg.sender];
    }

    function getaccess(uint number_hours) public returns (uint remainingBal) {
        uint total_fee = number_hours * hourly_fee;
        if (total_fee <= accounts[msg.sender]) {
            accounts[msg.sender] -= total_fee;
            owner.transfer(total_fee);
        }
        return accounts[msg.sender];
    }

    function withdraw(uint withdrawAmount) public returns (uint remainingBal) {
        if (withdrawAmount <= accounts[msg.sender]) {
            accounts[msg.sender] -= withdrawAmount;
            msg.sender.transfer(withdrawAmount);
        }
        return accounts[msg.sender];
    }

    function balance() public constant returns (uint) {
        return accounts[msg.sender];
    }

    function depositsBalance() public constant returns (uint) {
        require(msg.sender == owner, "only service owner may use this method");
        return address(this).balance;
    }
}
```

***Figure 2**: A simple contract for accessing an IoT service implemented in Solidity and EVM.*

As a conclusion smart contract allow us to:

- Turn legal obligations into automated processes;
- Guarantee a greater degree of security;
- Reduce reliance on trusted intermediaries;
- Lower transaction costs.

ADDRESSING BLOCKCHAIN CHALLENGES IN IOT

Integrating blockchain into IoT service architecture may come with flaws and shortcomings:

1. ***Scalability*** is one of the main roadblocks to the business adoption of blockchain technology. There are fears relating to the size of Blockchain ledger that might lead to centralization as it's grown over time and required some record management which is casting a shadow over the main quality of blockchain: decentralization. For example, Bitcoin can handle around 7 TPS (transactions per second), while Ethereum (the PoW version) executes around 20 TPS. These are considered extreme low throughputs for most business applications, not to mention the requirement to handle billions of transactions;

2. ***Processing power and time*** is required in order to encrypt all transaction data generated by the blockchain-based IoT ecosystem. This process is exhaustive given the fact that IoT ecosystems are very diverse and comprised of devices that have very different computing capabilities, especially those in the edge and gateway layers and not all of them will be capable of running the same encryption algorithms at the desired speed;

3. ***Storage*** is another roadblock. Blockchain's decentralized architecture implies that all its nodes are required to store the ledger and the ledger will increase in size as time passes. It the case of Bitcoin, the ledger size increases with almost 250MB per day. In a IoT-based ecosystems there are far more transactions and the storage should scale to accommodate terabytes of data on daily bases. This requirement cannot be fulfilled by most of the IoT devices and gateways.

These issues reside as part of the trade-off required to maintain complete decentralization (network partition) in a Brewer's theorem constrained environment, which implies that in the presence of a network partition, one has to choose between consistency and availability [24]. To go around these constraints, we need to make compromises on the decentralization, going with the assumption that we don't need global scale decentralization, but a local one. The solution is to implement a blockchains in blockhain architecture. An example is IoTeX [25], an auto-scalable and privacy-centric blockchain platform for the Internet of Things.

IoTeX

Instead of connecting all IoT nodes into one single blockchain, IoTeX groups the IoT nodes into subchains. In this regard, IoTeX is basically a network of many blockchains that are hierarchically arranged, where many blockchains can run concurrently with one another while retaining interoperability. At the top of the chain is the root blockchain, a public chain accessible by anyone, which has three main objectives:

1. Relay value and data across subchains in a privacy-preserving way to enable interoperability among subchains;
2. Supervision of subchains, e.g., penalize the bonded operators of subchain by bond confiscation;
3. Settlement and anchoring of payments and trust for subchains.

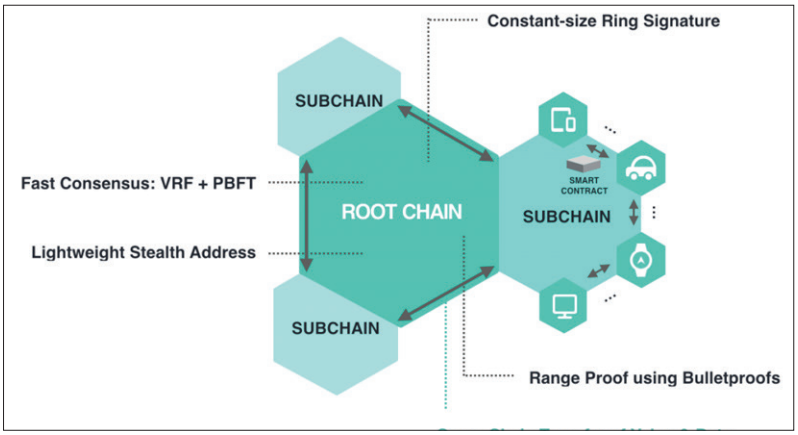

***Figure 3**: Iotex blockchains in blockchain, a rootchain and subchains architecture. (Source: iotex.io)*

IoTeX blockchain gives users the capability of provisioning their application-centric sub-chain, which is backed by the IoTeX root-chain, and on which they could do token transfer as well as execute smart contracts. It uses a proprietary DPoS consensus (Roll-DPoS), so that token holders could vote the delegates who will produce new blocks for the whole network. For block creation consensus, IoTeX is using Byzantine fault tolerance algorithm. IoTeX supports Ethereum virtual machine and smart contracts written in Solidity.

Properties	Rootchain	Subchain
Public vs Private	Public	Public or Private
Scalable	Required	Varies
Robust	Strongly Required	Required
Privacy-centric	Required	Varies
Extensibility	Non-Turing Complete	Turing Complete
Instant Block Finality	Required	Required

***Table 2**: Comparison Between Rootchain and Subchain*

HYPERLEDGER

Hyperledger [26] is a Linux Foundation project, which incubates and promotes a range of business blockchain technologies, including distributed ledger frameworks, smart contract engines, client libraries, graphical interfaces, utility libraries and sample applications.

Hyperledger could be used to develop the required blockchain building blocks, including consensus and smart contracts.

Some of the frameworks and tools Hyperledger provides are:

• ***Hyperledger Burrow*** is a permissionable smart contract machine, providing a modular blockchain client with a permissioned smart contract interpreter built in part to the specification of the Ethereum Virtual Machine;

• ***Hyperledger Fabric*** is a blockchain framework intended for developing applications or solutions with a modular architecture, allowing components, such as consensus and membership services, to be plug-and-play;

• ***Hyperledger Grid*** is a WebAssembly-based project for building supply chain solutions, seeking to assemble blockchain shared capabilities in order to accelerate the development of ledger-based solutions for all types of cross-industry supply chain scenarios;

• ***Hyperledger Sawtooth*** is a modular platform for building, deploying, and running distributed ledgers. Hyperledger Sawtooth includes a novel consensus algorithm, Proof of Elapsed Time (PoET) [27], which targets large distributed validator populations with minimal resource consumption;

• ***Hyperledger Iroha*** is a modular distributed blockchain platform with its own unique chain-based Byzantine Fault Tolerant consensus algorithm, called Yet Another Consensus and the BFT ordering service algorithms, rich role-based permission model and multi-signature support;

• ***Hyperledger Indy*** provides tools, libraries, and reusable components for creating and using independent digital identities rooted on blockchains or other distributed ledgers so that they are interoperable across administrative domains, applications and any other “silo”;

• ***Hyperledger Composer*** is a set of collaboration tools for building blockchain business networks that make it simple and fast for business owners and developers to create smart contracts and blockchain applications;

• ***Hyperledger Caliper*** is a blockchain benchmark tool, which allows users to measure the performance of a specific blockchain implementation with a set of predefined use cases;

• ***Hyperledger Explorer*** can view, invoke, deploy or query blocks, transactions and associated data, network information, chain codes and transaction families, as well as any other relevant information stored in the ledger.

CONCLUSIONS

The number of IoT devices is expected to grow with a steady 15% each year, a fact that is raising multiple issues regarding the security and data privacy. IoT is a vertical ecosystem with most of the devices being resource-constrained, unable to enforce complex security patterns. In this case the strategy to mitigate the security risks is to develop lightweight authentication and key agreements protocols.

For the rest of devices, mainly localized in the gateway and cloud computing layer, a decentralized digital ledger can be used to establish trust through the usage of smart contracts and blockchain technologies. In figure 2 we demonstrate a simple usage of smart contracts, implementing a Solidity contract to access data services, contract that can be deployed on Ethereum virtual machine.

We have also shown that a complete decentralized blockchain has extreme low throughput, unacceptable for large scale IoT deployment models. To mitigate this, we need to tradeoff some of the decentralization for faster consensus and validation. The solution is to implement a blockchains in blockchain architecture, where some of the workload is done in local clusters (subchains) with a high degree of autonomy by with standardized ways of communicating with the subchain network. A project that is doing just that is IoTeX, a faster and flexible IoT platform that is to be fully operational in the last quarter of 2019 and which supports Ethereum Virtual Machine and Solidity.

Besides IoTeX, there is Hyperledger, a Linux Foundation incubator for an extensive set of blockchain projects with focus on the re-use of common building blocks and rapid innovation of distributed ledger technologies and components. The Hyperledger provides with tools for implementing smart contracts and consensus, define business logic, develop deployment models and monitor performance.

ACKNOWLEDGMENT

This work was supported by the Institutional research programme "Competitiveness and efficiency through ICT specialization - SMARTIC" (2019-2022), project PN 1937-01-01 - "Research on advanced security policies and solutions for securing critical infrastructure against cyber-attacks", funded by the Ministry of Research and Innovation.

REFERENCES

[1] Knud Lasse Lueth. *State of the IoT 2018: Number of IoT devices now at 7B – Market accelerating*. 2018. URL: https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/ (visited on 03/04/2019).

[2] N. Miloslavskaya and A. Tolstoy. *Application of Big Data, Fast Data, and Data Lake Concepts to Information Security Issues*. In: 2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW). 2016, pp. 148–153. DOI: 10.1109/W-FiCloud.2016.41.

[3] Eran Toch, Yang Wang, and Lorrie Faith Cranor. *Personalization and Privacy: A Survey of Privacy Risks and Remedies in Personalization-based Systems*. In: User Modeling and User-Adapted Interaction 22.1-2 (Apr. 2012), pp. 203–220. ISSN: 0924-1868. DOI: 10.1007/s11257-011-9110-z. URL: http://dx.doi.org/10.1007/s11257-011-9110-z.

[4] J. Voelcker. *Stalked by Satellite - an Alarming Rise in GPS-enabled Harassment*. In: IEEE Spectr. 43.7 (Sept. 2006), pp. 15–16. ISSN: 0018-9235. DOI: 10.1109/MSPEC.2006.1652998. URL: https://doi.org/10.1109/MSPEC.2006.1652998.

[5] Andrew Odlyzko. *Privacy, Economics, and Price Discrimination on the Internet*. In: Proceedings of the 5th International Conference on Electronic Commerce. ICEC '03. Pittsburgh, Pennsylvania, USA: ACM, 2003, pp. 355–366. ISBN: 1-58113-788-5. DOI: 10.1145/948005.948051. URL: http://doi.acm.org/10.1145/948005.948051.

[6] Gregory L. Orgill et al. *The urgency for effective user privacy-education to counter social engineering attacks on secure computer systems*. In: SIGITE Conference. 2004.

[7] Joseph Menn. *Social networks scan for sexual predators, with uneven results*. URL: https://www.reuters.com/article/us-usa-internet-predators/social-networks-scan-for-sexual-predators-with-uneven-results-idUSBRE86B05G20120712 (visited on 03/02/2019).

[8] Jan Henrik Ziegeldorf, Oscar García Morchon, and Klaus Wehrle. *Privacy in the Internet of Things: Threats and Challenges*. In: CoRR abs/1505.07683 (2015). arXiv: 1505.07683. URL: http://arxiv.org/abs/1505.07683.

[9] D. M'Raihi et al. HOTP: *An HMAC-Based One-Time Password Algorithm*. RFC 4226. RFC Editor, 2005. DOI: 10.17487/RFC4226. URL: https://www.rfc-editor.org/info/rfc4226.

[10] D. M'Raihi et al. TOTP: *Time-Based One-Time Password Algorithm*. RFC 6238. RFC Editor, 2011. DOI: 10.17487/RFC6238. URL: https://www.rfc-editor.org/info/rfc6238.

[11] Rob Price. *Hackers stole millions of Facebook users' highly sensitive data — and the FBI has asked it not to say who might be behind it*. URL: https://www.businessinsider.com/facebook-30-million-users-affected-hack-fbi-asked-not-to-reveal-source-2018-10 (visited on 10/12/2018).

[12] Douglas MacMillan and Robert McMillan. *Google Exposed User Data, Feared Repercussions of Disclosing to*

Public. URL: https : / / www . wsj . com / articles / google - exposed - user - data - feared - repercussions-of-disclosing-to-public-1539017194 (visited on 10/08/2018).
[13] Douglas MacMillan and Robert McMillan. *Quora says 100 million users may have had their account information stolen in a massive data breach*. URL: https://www.businessinsider.com/quora- says-it-had-massive-hack-that-may-have-affected100-million-users-2018-12 (visited on 12/04/2018).
[14] Paige Leskin. *Here's how to check if you were one of the 500 million customers affected by the Marriott hack*. URL: https://www.businessinsider.com/marriott-starwood-hotel-hack-data-breach-how-to-check-if-you-were-affected -2018-11 (visited on 11/30/2018).
[15] Q. Jiang et al. *Lightweight three-factor authentication and key agreement protocol for internet-integrated wireless sensor networks*. In: IEEE Access 5 (2017), pp. 3376–3392. ISSN: 2169-3536. DOI: 10.1109/ACCESS. 2017.2673239.
[16] Radu Boncea and Ioan Bacivarov. *A System Architecture for Monitoring the Reliability of IoT*. In: Proceedings of the 15th International Conference on Quality and Dependability. SOCIETATEA ROMÂNĂ PENTRU ASIGURAREA CALITĂȚII. 2016, pp. 143–150.8
[17] Ahmed Banafa. *Secure and Smart Internet of Things (IoT). Using Blockchain and AI*. River Publishers Series in Information Science and Technology, 2018. ISBN: 9788770220309.
[18] Markus Jakobsson and Ari Juels. *Proofs of Work and Bread Pudding Protocols*. In: Proceedings of the IFIP TC6/ TC11 Joint Working Conference on Secure Information Networks: Communications and Multimedia Security. CMS '99. Deventer, The Netherlands, The Netherlands: Kluwer, B.V., 1999, pp. 258–272. ISBN: 0-7923-8600-0. URL: http://dl.acm.org/citation.cfm?id=647800.757199.
[19] Mthcl. *The math of Nxt forging*. Tech. rep. Nxt Community, 2014. URL: https://www.docdroid.net/e29h/forging0-5-1.pdf.
[20] Dan Larimer. *DPOS Consensus Algorithm - The Missing White Paper*. URL: https://steemit.com/dpos/@dantheman/dpos-consensus-algorithm-this-missing-white-paper (visited on 03/02/2019).
[21] Miguel Castro and Barbara Liskov. *Practical Byzantine Fault Tolerance*. In: Proceedings of the Third Symposium on Operating Systems Design and Implementation. OSDI '99. New Orleans, Louisiana, USA: USENIX Association, 1999, pp. 173–186. ISBN: 1-880446-39-1. URL: http://dl.acm.org/citation.cfm?id=296806.296824.
[22] Brian Curran. *What is Practical Byzantine Fault Tolerance?* URL: https://blockonomi.com/practical-byzantine-fault-tolerance/ (visited on 03/02/2019).
[23] Chris Dannen. *Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners*. 1st. Berkely, CA, USA: Apress, 2017. ISBN: 1484225341, 9781484225349.
[24] Seth Gilbert and Nancy Lynch. *Brewer's Conjecture and the Feasibility of Consistent, Available, Partitiontolerant Web Services*. In: SIGACT News 33.2 (June 2002), pp. 51–59. ISSN: 0163-5700. DOI: 10.1145/564585.564601. URL: http://doi.acm.org/10.1145/564585.564601.
[25] The IoTeX Team. *IoTeX. A Decentralized Network for Internet of Things Powered by a Privacy-Centric Blockchain*. White Paper. 2018. URL: https://s3.amazonaws.com/web-iotex-static/home/IoTeX_Whitepaper_1.5_EN.pdf.
[26] The Linux Foundation. *Hyperledger Business Blockchain Technologies*. URL: https://www.hyperledger.org (visited on 03/02/2019).
[27] Lin Chen et al. *On Security Analysis of Proof-of-Elapsed-Time (PoET)*. In: Stabilization, Safety, and Security of Distributed Systems. Ed. by Paul Spirakis and Philippas Tsigas. Cham: Springer International Publishing, 2017, pp. 282–297. ISBN: 978-3-319-69084-1.
[28] Z. Zheng et al. *An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends*. In: 2017 IEEE International Congress on Big Data (BigData Congress). 2017, pp. 557–564. DOI: 10.1109/BigDataCongress.2017.85.