Article

EconLedger: A Proof-of-ENF Consensus based Lightweight Distributed Ledger for IoVT Networks

Ronghua Xu, Deeraj Nagothu and Yu Chen *

Dept. of Electrical and Computer Engineering, Binghamton University, SUNY, Binghamton, NY 13905, USA; rxu22@binghamton.edu (R.X.); dnagoth1@binghamton.edu (D.N.)

* Correspondence: ychen@binghamton.edu; Tel.: +1-607-777-6133

Abstract: The rapid advancement in artificial intelligence (AI) and wide deployment of Internet of Video Things (IoVT) enable situation awareness (SAW). Robustness and security of the IoVT systems are essential to a sustainable urban environment. While blockchain technology has shown great potentials to enable trust-free and decentralized security mechanisms, directly embedding crypto-currency oriented blockchain schemes into resource-constrained Internet of Video Things (IoVT) networks at the edge is not feasible. Leveraging Electrical Network Frequency (ENF) signals extracted from multimedia recordings as region-of-recording proofs, this paper proposes EconLedger, an ENF-based consensus mechanism that enables secure and lightweight distributed ledgers for small scale IoVT edge networks. The proposed consensus mechanism relies on a novel Proof-of-ENF (PoENF) algorithm where a validator is qualified to generate a new block if and only if a proper ENF-containing multimedia signal proof is produced within the current round. Decentralized database (DDB) is adopted to guarantee efficiency and resilience of raw ENF proofs on the off-chain storage. A proof-of-concept prototype is developed and tested in a physical IoVT network environment. The experimental results validated the feasibility of the proposed EconLedger to provide a trust-free and partially decentralized security infrastructure for IoVT edge networks.

Keywords: Electrical Network Frequency (ENF), Proof-of-ENF (PoENF), Consensus, Blockchain, Security, Internet of Video Things (IoVT).

1. Introduction

Thanks to the rapid advancements in artificial intelligence (AI) and Internet of Things (IoT) technologies, the concept of Smart Cites becomes realistic. The information fusion capability provided by these interconnected devices enables situation awareness (SAW), which is essential to ensure a safe and sustainable urban environment. With wide deployment of the exponentially increasing smart Internet of Video Things (IoVT) for safety surveillance purposes, intelligent online video streams processing becomes one of the most actively researched topics in Smart Cites [1].

In typical Internet of Video Things (IoVT) systems, a huge amount of raw video data collected by geographically scattered cameras is sent to a remote cloud for aggregation. It provides a broad spectrum of promising applications, including public space monitoring, human behavior recognition [2], and suspicious event identification [3]. However, centralized IoVT solutions suffer from the risk of single points of failure and are not scalable to accommodate the ever growing IoVT networks, which are pervasively deployed with heterogeneous and resource-limited smart devices at the edge of networks. Moreover, online video streams and other offline data, like situation contextual features, are shared among participants through high-end cloud servers, which are under the control of third-party entities. Such a centralized architecture also raises severe privacy and security concerns that data in storage can be misused or tampered with by dishonest entities.

Evolving from the distributed ledger technology (DLT), blockchain has gained significant attentions for its potential to revolutionize multiple areas of economy and society. The inherent security guarantees of blockchain lay down the foundations of a serverless record-keeping without the need for centralize trusted third-party authorities [4]. Blockchain runs on a decentralized peer-to-peer (P2P) network to securely store and verify data without relying on a centralized trust authority. The decentralization removes the risk of singular point of failures and mitigate the bottleneck performance, which were inherent in centralized architectures. In addition, blockchain leverages distributed consensus protocols to enable a verifiable process for fault tolerance and tamper-proof storage on a public distributed ledger. Therefore, transparency, immutability and auditability guaranteed by blockchain ensure resilience, correctness, and provenance for all data sharing among untrusted participants.

Internet of Video Things (IoVT) provides a broad spectrum of applications, particularly in the area of public safety [5]. Migrating from centralized cloud-based paradigms to decentralized blockchain-based methods makes IoVT systems more efficient, scalable and secure. However, directly integrating cryptocurrency-oriented blockchains into resource constrained IoVT systems is difficult to handle the blockchain trilemma [6], which points out that decentralization, scalability and security cannot perfectly co-exist. Most IoVT devices are highly resource constrained. Therefore, computing and storage intensive consensus protocols are not affordable, like Proof-of-Work (PoW) [7], Proofs-of-Retrievability (PORs) [8], or Practical Byzantine Fault Tolerant (PBFT) [9], which is with high communication complexity and poor scalability. In addition, IoVT systems involve a large volume of real-time transactions. Higher throughput and lower latency become key metrics in blockchain-based systems for IoVT deployed on edge networks. Furthermore, DLTs are not general-purpose databases. The storage overhead is prohibitively high if raw data generated by IoVT transacting networks are stored in the blockchain.

Electrical Network Frequency (ENF) is the power supply frequency which fluctuates around its nominal frequency (50/60Hz). The frequency fluctuations vary based on the geographical region. The ENF fluctuations estimated from simultaneously recorded audio/video recordings within a power grid have a high correlation similarity [10].

Inspired by spatio-temporal sensitive ENF contained in multimedia signals, this paper proposes *EconLedger*, a novel *Proof-of-ENF* (PoENF) consensus algorithm based lightweight DLT for small scale IoVT networks. Compared to PoW or PoRs, which needs high computation or storage resources in mining process, our novel PoENF consensus requires each validator to use extracted ENF variations from simultaneous multimedia recordings as proofs during current consensus round. The validator who presents a valid ENF proof with minimal squared-distance-based score is qualified to generate a new block. Thus, PoENF consensus mechanism not only achieves efficiency without high demand of mining resource or hardware platform support, it also enhances security by mitigating mining centralization.

In contrast to existing solutions that directly collect ENF fluctuations from power grids and stores audio/video recordings in a centralized location-dependent ENF database [10,11], EconLedger uses *Swarm* [12], a decentralized database (DDB) technology, to archive raw ENF-containing multimedia proofs and transactions over IoVT networks. Only hashed references of data are recorded on an immutable and auditable distributed ledger. Thus, it reduces the ever-increasing data storage overhead on the public ledger. The EconLedger ensures correctness, availability and provenance of data sharing among untrust devices under a distributed network environment. Moreover, a permissioned network ensures that only authorized nodes can access raw data on DDB, such that privacy preservation is guaranteed.

In summary, this paper makes the following contributions:

1) A secure-by-design EconLedger architecture is introduced along with a detailed explanation of the key components and work flows;

2) A novel PoENF consensus mechanism is proposed that improves the resource efficiency and achieves a higher throughput than PoW-based blockchains do;
3) A finalized on-chain ledger is coupled with a decentralized off-chain storage to resolve storage burden, and it guarantees security and robustness of data sharing and cooperation in IoVT networks; and
4) A proof-of-concept prototype is implemented and tested on a small scale IoVT network, and experimental results verified that the EconLedger is feasible and affordable to the IoVT devices deployed at edge networks.

The remainder of this paper is organized as follows: Section 2 briefly discusses background knowledge of ENF, then reviews existing consensus algorithms and state-of-the-art research on IoT-Blockchains. Section 3 introduces the rationale and architecture of EconLedger, as well as core features and security guarantees. A novel PoENF consensus mechanism is explained in Section 4. Section 5 presents prototype implementation and numerical results, and discusses performance improvements and security insurances. Finally, a summary is presented in Section 6.

2. Background and Related Work

2.1. ENF as a Region-of-Recording Fingerprint

ENF is the supply frequency in power distribution grids, which has a nominal frequency of 50Hz or 60Hz depending on the location of the power grids. Due to environmental effects in the grid such as load variations and control mechanisms, the instantaneous ENF usually fluctuates around its nominal value. At a given time, variation trends of ENF fluctuations from all locations of the same grid are almost identical due to the interconnected nature of the grid [13]. ENF fluctuations are embedded in audio/video recordings either due to the electromagnetic induction or background hum from devices connected to the power grid [14]. Thanks to the consistency and reliability of ENF at a time instant, ENF has been adopted as a forensic tool to identify forgeries in multimedia recordings. All ENF signals estimated from simultaneous multimedia recordings at different locations have similar fluctuations throughout the power grid. Thus, there are multiple forensic applications based on ENF, such as validating the time-of-recording of an ENF-containing multimedia signal [14] and estimating its location-of-recording [15].

In IoVT systems, ENF signals extracted from video recordings are in the form of illumination frequency (120Hz). The video recordings made under indoor artificial light include ENF fluctuations. The estimation of ENF signals depends on the type of imaging sensor used in a camera. The most commonly used imaging sensors are complementary metal oxide semiconductor (CMOS) and charge-coupled devices (CCD) sensors, which have different shutter mechanisms. In this work, we assume that ENF signals are extracted from video recordings generated by cameras with CMOS imaging sensors in an indoor setting with artificial light [11,16]. Estimation of ENF involves various signal processing techniques, like power spectral analysis and spectrogram-based techniques, which are beyond the scope of this paper.

2.2. Consensus Protocols for Blockchain

2.2.1. Nakamoto Protocols

The Nakamoto protocol is implemented as the consensus foundation of Bitcoin [7], and it is widely adopted by many cryptocurrency-based blockchain networks like Ethereum [17]. The Nakamoto protocol adopts a computation-intensive PoW, which requires all participants compete for rewards through a cryptographic block-hash value discovery racing game. The PoW consensus demonstrate security and scalability in an asynchronous open-access network as long as an adversary does not control the majority (51%) of the miners. However, the brute-force PoW mining process also incurs a high demand of computation and energy consumption such that it is not affordable on resource constrained IoT devices.

To improve the performance and resource usage efficiency in PoW, a number of alternative Proof of X-concept (PoX) schemes have been proposed. Permacoin [8] repurposes mining resources in PoW to achieve a distributed storage of archival data. The Permacoin adopts PORs [18], which requires miners to present random access to a copy of a file from local storage as valid proofs for successfully minting money. Permacoin requires participants to invest its storage capacity rather than solo computational power. It could reduce unnecessary wastage of computational resources in PoW and mitigate centralized mining pools issue.

Similar to Permacoin, a Resource-Efficient Mining (REM) [19] scheme is proposed to achieve security and resource efficiency based on the partially decentralized trust model inherent in Intel Software Guard Extensions (SGX). The REM utilizes a Proof-of-Useful-Work (PoUW) consensus protocol, which requires miners provide trustworthy measuring on CPU cycles used by its useful workloads in SGX-protected enclave. Compared with Proof-of-Elapsed-Time (PoET) in Sawtooth [20] that uses random idle CPU time as proofs, PoUW in REM not only prevents against the stale chip problem, but also yields the smallest amount of mining waste.

To reduce energy consumption caused by intensive hash value calculating in PoW, Peercoin [21] adopts Proof-of-Stake (PoS), which leverages the distribution of token ownership to simulate a verifiable random function to propose new blocks. Such a process of efficient "virtual mining" manner allows PoS miners only consume limited computational resources to generate new blocks. Similar to PoW, PoS guarantees security as long as an adversary owns no more than half of the total stakes in the network.

Unlike PoW and its variants, the PoENF consensus scheme neither requires high demand of computation and storage for mining, nor depends on security guarantees supported by trusted hardware or monetary deposit stake. It is suitable for heterogeneous IoVT devices connected to the power grid.

2.2.2. Byzantine Fault Tolerant Protocols

As the first practical BFT consensus, PBFT [9] uses the State Machine Replication (SMR) scheme to address the Byzantine General Problem [22] in distributed networks. It has been widely adopted as a basic consensus solution in the permissioned blockchains, like Hyperledger Fabric [23]. The PBFT algorithm guarantees both liveness and safety in synchronous network environments if at most $\lfloor \frac{n-1}{3} \rfloor$ out of total of n replicas are Byzantine faults. Compared to those probabilistic Nakamoto blockchains, BFT-based consensus networks ensure a deterministic finality on distributed ledger. However, it inevitably incurs high latency and communication overhead as synchronously executing consensus protocol among all nodes in large scale networks.

Therefore, combining Nakamoto-style block generation with BFT-style chain finality provides a prospective solution to ensure data consistency and immediate finality. Casper [24] introduces a lightweight chain finality layer on top of a Nakamoto protocol, like PoW and PoS. In Casper, a fixed set of validators execute a PoW block proposal protocol to maintain an ever-growing *block tree*, while an efficient voting-based process is responsible to commit a direct ancestor block of the finalized parent block as a *checkpoint*. Finally, only a unique checkpoint block path from checkpoint tree is accepted as the finalized chain.

Unlike Casper, which is a PoS-based finality system overlaying an existing PoW blockchain, our EconLedger uses a voting-based chain finality to resolve the forks caused by probabilistic PoENF block generation.

2.3. State of the Art on IoT-Blockchain

To support security and lightweight features required in IoT systems, IoTChain [25] proposes a three-tier blockchain-based IoT architecture, which allows regional nodes to perform any lightweight consensus, like PoS and PBFT. IoTChain only provides simulation results on communication cost of transactions; however key metrics in

the consensus layer, like computation, storage and throughput, are not considered. FogBus [26] proposes a lightweight framework for integrating blockchain into fog-cloud infrastructure, which aims to ensure data integrity as transferring confidential data over IoT-based systems. In FogBus, master nodes deployed at the fog layer are allowed to perform PoW mining, while IoT devices send transactions to master nodes as trust intermediates to interact with blockchain. However, using PoW as the backbone consensus protocol still results in high energy consumption and low throughout.

HybridIoT [27] proposes a hybrid blockchain-IoT architecture to improve scalability and interoperability among sub-blockchains. In HybridIoT, a BFT inter-connector framework works as a global consortium-blockchain to link multiple PoW sub-blockchains. However, using PoW consensus in sub-blockchain networks still brings computation and storage overhead on IoT devices if they are deployed as full nodes. IoTA [28] aims to enable a cryptocurrency designed for the IoT industry, and it leverages a directed acyclic graph (DAG), called tangle [29], to record transactions rather than chained structure of the ledger. IoTA provides a secure data communication protocol and zero fee micro-transaction for IoT/machine-to-machine (M2M), and it demonstrates high throughput and good scalability. However, existing IoTA networks still rely on hard-coded coordinators, which employ PoW to finalize path of recorded transactions in DAG.

Unlike the above mentioned IoT-Blockchain solutions, which either adopt computation intensive PoW as their backbone consensus mechanism or rely on a intermediate fog layer to execute consensus protocol, EconLedger aims to provide a partially decentralized and lightweight blockchain for resource constrained IoVT devices at the edge without relying on any intermediate consensus layer deployed at fog level. Moreover, EconLedger leverages a DDB technology to enable a trust off-chain storage, which reduces storage overhead caused by directly storing raw data on the public distributed ledger.

3. EconLedger: Rationale and Architecture

3.1. System Design Overview

EconLedger aims at a secure-by-design, trust-free and partially decentralized infrastructure for cross-devices networking IoVT systems at the edge. We consider a small scale video surveillance network with 100 nodes, and all IoVT devices and edge/fog servers are connected to the same regional power grid. Here, a node refers to a device owned by an user. Figure 1 is the system architecture of EconLedger.

3.1.1. IoVT Application

The upper-level IoVT application utilizes a EconLedger fabric to enable a decentralized video analytic service and information visualization at the edge. All devices and users must be registered to join the IoVT system as required by the permissioned network, which can provide basic security primitives, like public key infrastructure (PKI), identity authentication [30] and access control [31], etc.. Real-time video streams generated by cameras are transferred to on-site/near-site edge devices for lower level analytic tasks, like object-detection and situational contextual features extraction. Thus, cameras associating with edge devices act as IoVT service units at the network of edge. Then, IoVT service units send raw video data and extracted contextual information to information visualization unit, which provides video recordings and smart applications for authorized users.

To prevent visual layer attacks, IoVT service extracts ENF signals from video streams as an environmental fingerprint, which is stored into DDB secured by *EconLedger fabric*. At any given time instant, variation trends of ENF-containing multimedia signals from all synchronous cameras on the same power grid are almost identical. Therefore, using ENF fluctuations recorded on EconLedger laid a solid ground truth for video authenticity verification. Through calculating correlation coefficients among ENF signals extracted

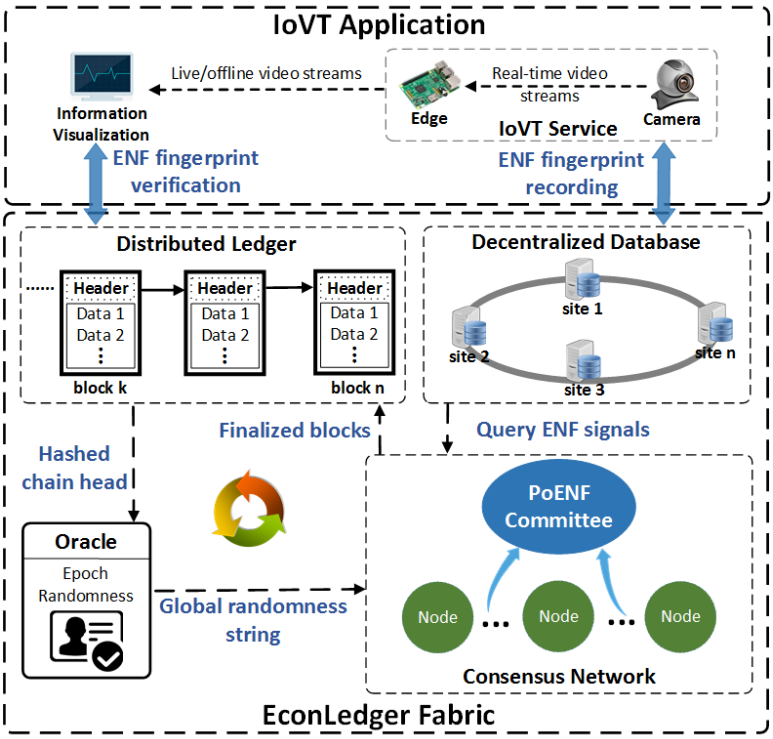

Figure 1. The EconLedger system architecture.

from video recordings with an agreed ENF estimate recorded on distributed ledger, information visualization unit verifies whether or not live/offline video streams are generated by cameras within the same power grid [32,33].

3.1.2. EconLedger Fabric

The EconLedger fabric provides a fundamental networking and security infrastructure to support decentralized security features for the IoVT system. All authorized devices firstly store raw ENF fingerprints into the DDB, then launch transactions that include hashed references of raw data along with valid signatures. As transactions store fixed length of hashed references rather than raw data with varying size, such an off-chain manner reduces storage overhead when IoVT devices verify transactions and synchronize the ever-increasing distributed ledger.

EconLedger uses a small PoENF committee to achieve high efficiency by reducing messages propagation delay and communication overhead on the edge network. Given a random committee election mechanism, only a subset of nodes within the network are elected as PoENF committee members. The PoENF consensus protocol is only executed by validators of a PoENF committee instead of all nodes in the network. Therefore, the scalability is improved at the cost of partially decentralization by a PoENF consensus committee.

Meanwhile, a random PoENF committee rotation strategy ensures that the robustness is not sacrificed due to fewer validators. Combining the current status of the distributed ledger, a distributed randomness protocol acts as the oracle to periodically generate global randomness strings for PoENF committee selection. As randomness strings are bias-resistant and unpredictable, the probability of an adversary dominating a subsequent committee is decreased exponentially even if the current PoENF committee is compromised.

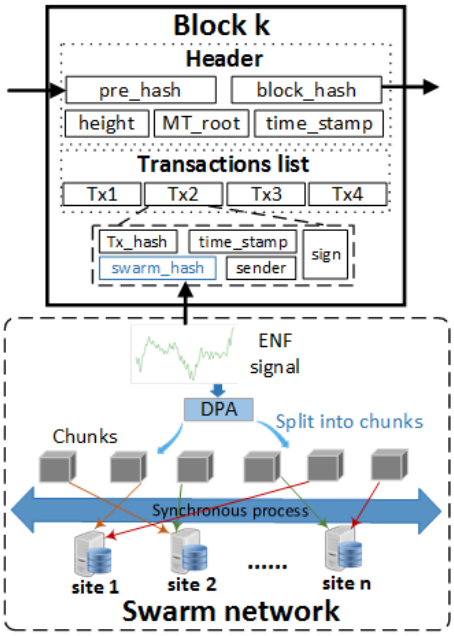

Figure 2. The illustration of block & off-chain data structure.

3.2. Network Model

EconLedger relies on a permissioned network, and we assume that the system administrator is a trust oracle to maintain global identity profiles for all valid nodes. We adopt a standard asymmetrical algorithm like Rivest–Shamir–Adleman (RSA) for key generation ($RSA.gen$) and digital signature scheme ($RSA.sign$, $RSA.verify$). During the registration process, a signing-verification keys pair $(sk_i, pk_i) \leftarrow RSA.gen(i)$ is generated by PKI and assigned to the authorized node u_i. Additionally, a node's public key pk_i is associated with its credit stake $c_i \leq C_{max}$, where C_{max} is the maximum value of credit stake defined by system. Therefore, all registered nodes can be represented as $U = \{(pk_1, c_1), (pk_2, c_2), ..., (pk_n, c_n)\}$, where n is the total number. As above security assumptions depend on the system administrator's behavior, our EconLedger is a partially decentralized blockchain model.

EconLedger assumes a synchronous network environment. Operations in consensus protocol are coordinated in rounds with upper bounded delay $\mathcal{T}_\Delta$. Thus, the time is divided into discrete *slots*, which can be indexed by logical clocks *ticks* to synchronize the events in a distributed system [34]. Given a certain tick $t \in \{1, 2, 3, ...\}$, slot sl_t represents the length of time window to measure $\mathcal{T}_\Delta$. Time window of sl_t should be sufficient to guarantee that message transmitted by a sender is received by its intended recipients (accounting for local time discrepancies and network delays). Thus, we require that $sl_t \geq \mathcal{T}_\Delta$ to ensure liveness of consensus protocol.

3.3. Hybrid On-chain and Off-chain Storage

To address issues of high storage overhead incurred by directly saving raw data into DLTs, EconLedger utilizes a hybrid on-chain and off-chain storage solution. Figure 2 illustrates the block & off-chain data structure used in EconLedger. The block is the basic unit of on-chain storage, which includes block header and the orderly transactions list. The *MT_root* in block header stores the hash root of Merkle tree to maintain the integrity of all transactions. In each transaction, the *swarm_hash* only stores references to the data rather than the data themselves. As references are hash values with fixed length like 32 or 64 bytes, all transactions have almost the same size even if linked raw data have large size or require different formats, like ENF signals or multimedia recordings.

Table 1: Relevant Basic Notation.

Symbol	Description
sl_E	Epoch including sequential order of time slot
D	Dynasty represents current PoE committee
tx	A transaction broadcasted by the node of network
B	A block proposed by the validator in current Dynasty
$\mathcal{C}$	Distributed ledger maintained by the consensus network

The off-chain storage relies on a Swarm network, in which all sites cooperatively construct a DDB system. In EconLedger, a site refers to a fog/edge server. The data uploaded to Swarm are cut into pieces called *chunks*, which is the basic unit of storage and retrieval in the Swarm network. Each chunk can be accessed at a unique address which is calculated by its hashed content. All data chunks use their chunk hash to construct a Merkel hash tree, which root is the reference to retrieve raw data. Swarm implements a specific type of content addressed distributed hash tables (DHTs), called Distributed Pre-image Archive (DPA), to manage chunks across distributed sites. All Swarm sites have their own base addresses with the same size as the chunk hash, and the site(s) closest to the address of a chunk do not only serve information about the content but actually host the data [35]. All sites in the Swarm network use Kademlia DHT protocol [36], which synchronizes chunks in a P2P manner, to ensure data persistence and redundancy.

4. PoENF: a Proof-of-ENF Consensus Protocol

4.1. Basic Notation

Table 1 describes relevant notation used in PoENF model. To model sequential events in synchronous consensus rounds, a set of sub-sequential slots are used to define $Epoch$, which is represented as $sl_E = \{sl_1, sl2, ..., sl_t\}$, where $0 \leq t \leq R$, and epoch size R is a value of multiple unit slot sl. A *validator* $v_i \in V$ ($V \subseteq U$) is a valid node who is qualified for being selected as a PoENF committee member. We define *Dynasty* to represent current PoENF committee, which is denoted as $D = \{(pk_1, c_1), (pk_2, c_2), ..., (pk_k, c_k) \subseteq V\}$, where $0 \leq k \leq K$, and K is the PoENF committee size. We use $\mathcal{H}(\cdot)$ to denote a pre-defined collision-resistant hash function that output hash string $h \in \{0,1\}^\lambda$.

Before introducing key features and components in the PoENF consensus protocol, several basic definitions are introduced as following:

Definition 1. *Transaction - is launched by a node u_i and represented as $tx = \{tx\_hash, pk_i, T_{stamp}, data, \sigma_i\}$, where:*

- *tx_hash: is a λ-bit-length hash string of transaction tx, which is calculated by $\mathcal{H}(pk_i, T_{stamp}, data)$;*
- *pk_i: is sender's public key;*
- *T_{stamp}: is time stamp of generating transaction;*
- *$data$: is the information $d \in \{0,1\}^*$ enclosed by the transaction, like swarm_hash or any byte strings; and*
- *σ_i: is a signature $RSA.sign_{sk_i}(tx\_hash, pk_i, T_{stamp}, data)$ signed by sender's private key sk_i.*

Definition 2. *Block - generated at slot sl_t ($t \in 1, 2, 3, ..$) by validator v_j is represented as $B_i = (pre\_hash, height, mt\_root, tx\_list, sl_t, pk_j, \sigma_j)$, where*

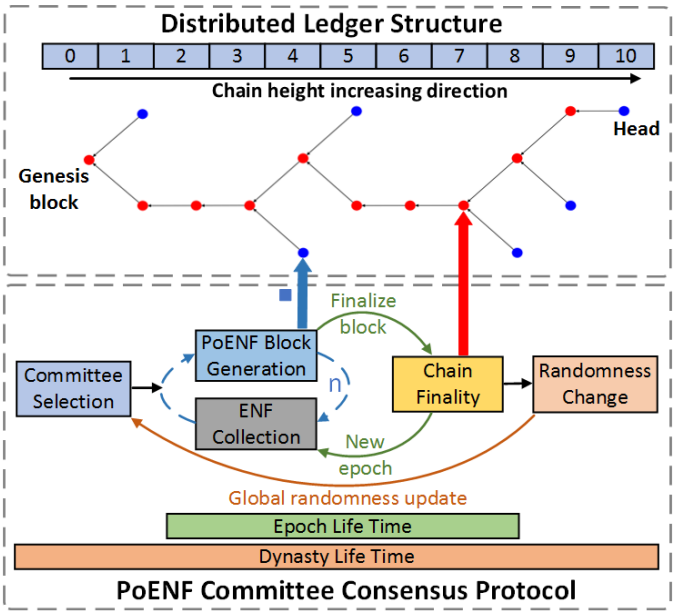

Figure 3. The PoENF consensus protocol overview.

- *pre_hash: is a λ-bit-length hash string of previous Block B_{i-1}, which is calculated by $\mathcal{H}(pre\_hash, height, mt\_root, tx\_list, sl_{t-1})$;*
- *height: is height of current block in blockchain (ledger);*
- *mt_root: is a root hash of Merkle tree of tx_list;*
- *tx_list: is an orderly transactions list $[tx_1, tx_2, ..., tx_n]$;*
- *sl_t: is block created time stamp at the round sl_t;*
- *pk_j: is public key of validator v_j; and*
- *σ_j: is a signature $RSA.sign_{sk_j}(pre\_hash, height, mt\_root, tx\_list, sl_t, pk_j)$ signed by validator v_j.*

We define a special block called *Genesis Block*, that is represented as $B_0 = (pre\_hash = 0, height = 0, sl_0 = 0, init\_D)$, where $init\_D$ is the initial dynasty. Therefore, all on-chain data on the distributed ledger is organized as an ordered sequence of blocks started from B_0.

Definition 3. *Blockchain (Distributed Ledger) - is a partial order of blocks, that is represented as $\mathcal{C} = B_0 \rightarrow B_1 \rightarrow ... \rightarrow B_{n-1} \rightarrow B_n$ indexed by strictly increasing slots sl_t. Each block B_i uses its pre_hash=$\mathcal{H}(B_{i-1})$ to link with the previous block B_{i-1}, and key parameters are:*

- *length: the length of the chain denoted $len(\mathcal{C}) = n$ to count the number of blocks between the genesis block B_0 and the confirmed block B_n; and*
- *head: the head of the chain denoted $head(\mathcal{C}) = B_n$, where B_n is the last confirmed block that is extended on finalized main chain.*

4.2. PoENF Committee Consensus Protocol: Overview

Figure 3 is an overview of the PoENF consensus protocol, which includes the distributed ledger structure and PoENF committee consensus workflows. The distributed ledger in EconLedger follows a tree structure originated from the genesis block. Each new block extends its chain path through *pre_hash* to point to a parent block. All nodes in such a ledger tree can be represented as confirmed blocks (blue) or finalized blocks (red), as the upper part of Figure 3 shows. The chain height follows a strictly increasing sequence of finalized blocks, therefore, a valid path can only go through those red nodes. The head of a blockchain is anchored on a recently confirmed block which has linked to a finalized chain with the largest height value.

At the configuration stage, the system administrator specifies a group of validators as the initial PoENF committee to initialize an EconLedger network. The lower part of Figure 3 demonstrates workflows of the PoENF committee consensus protocol including the dynasty cycle and the epoch cycle. A dynasty cycle starts from committee selection and ends when the global randomness string of current dynasty has been updated by the randomness change process. At the beginning of a dynasty's lifetime, the committee selection process uses the current global randomness string as a seed for committee election protocol, which exploits a Verifiable Random Function (VRF) based cryptographic sorting scheme [37]. Given credit weights of all nodes, K validators are randomly chosen to construct an new PoENF committee D, which will be added to the current block. Finally, validators of new D establish a fully connected P2P consensus network and start a new dynasty cycle.

For each epoch cycle, PoENF block generation and chain finality are core functions that ensure the liveness and termination in continuous consensus rounds. By calculating a squared-distance of ENF proofs from validators, the PoENF mechanism determines that a validator with the minimal squared-distance based score can generate a valid block in current block proposal round. After n rounds of block proposal, chain finality relies on a voting-based chain finality mechanism to resolve fork issues caused by conflicting confirmed blocks, and finalizes the history of ledger data through a unique chain path.

At the end of current dynasty, PoENF committee members utilize the RandShare mechanism to cooperatively reach the agreement on proposing a new global randomness string. As a distributed randomness protocol, RandShare adopts Publicly Verifiable Secret Sharing (PVSS) [38] to ensure unbiasability, unpredictability, and availability in public randomness sharing. The proposed unbiasable and unpredictable global randomness string will be updated as the new seed for the committee selection process of sub-sequential dynasty.

4.3. PoENF-based Block Proposal Mechanism

The PoENF-based block proposal mechanism is mainly responsible for generating candidate blocks and extending them along a finalized chain path. Following principles of chain based Nakamoto protocols, the PoENF algorithm simulates a virtual mining manner by pseudorandomly specifying a validator of committee as the slot leader to generate block. To generate a block, a validator must present an ENF proof which has the minimum squared-distance score in current round. All honest validators accept valid blocks and ensure that only one block is extended on the finalized main chain of local distributed ledger.

4.3.1. Transactions Pooling

Given a certain period of sliding window for ENF collection, each validator collects transactions from valid nodes. If a transaction stores reference that point to ENF proof, it is an ENF proof transaction. In current block generation round, each node is required to send only one ENF proof transaction. After receiving the broadcasted transactions, each validator verifies buffered transactions according to pre-defined conditions:

i) Transaction sender $u_i \in U$, and $RSA.verify(tx\_hash, pk_i, T_{stamp}, data) = \sigma_i$ by using the sender's public key pk_i;
ii) ENF proof tx sent by u_i should not be existed in the transactions pool; and
iii) Time stamp T_{stamp} must fall into current time slot;

The condition i) prevents transactions from invalid nodes or any malicious modification. While conditions ii) and iii) are mainly for preventing duplicated ENF proof tx in current period of time slot. After the verification process, only valid transactions are cached as local transactions pool denoted as $TX = \{tx_1, tx_2, ..., tx_N\}$, where N is the transactions pool size. The validator also uses condition iii) to regularly check local transaction pool and removes outdated transactions that has not been recorded in the latest confirmed block.

4.3.2. PoENF Consensus Algorithm

Given current transactions pool TX, the validator $v_i \in D$ chooses all ENF proof transactions generated by committee members to construct a ENF proof transactions list $TX_{ENF} = \{tx_1, tx_2, ..., tx_K\}$, where K is the committee size. An ENF proof is a vector $E = \{e_1, e_2, ..., e_d\}$, where $e_i \in \mathbb{R}$ is the ENF sample value and d is the samples size. By using *swarm_hash* that is stored in *data* parameter of tx_k, the E_k sent by v_k can be fetched from off-chain storage. Thus, each v_i can locally maintain a set of collected ENF proof vectors $G_i = \{E_1, E_2, ..., E_k\}$, where $k \leq K$.

To become a slot leader and propose a new block in current block proposal round, v_i must show that its ENF proof E_i can solve a PoENF puzzle problem. Intuitively, the goal of PoENF puzzle problem is to choose an E_k that deviates the least from all ENF proofs in G_i based on their relative distances, which are computed with the Euclidean norm. However, a single Byzantine validator can force the PoENF algorithm to choose any arbitrary ENF proof by sending a poisoned E_b that is too far away from other ENF proofs. Therefore, our PoENF algorithm adopts Krum aggregation rule to provide an (α, f)-Byzantine resilience property [39].

For each $v_i \in D$, let $G_i = \{E_1, E_2, ..., E_n\}$ include $n \geq 2f + 3$ collected ENF proofs from PoENF committee members, and only at most f are sent by Byzantine nodes. For any $i \neq j$, let $i \to j$ denote the fact the that E_j belongs to the $n - f - 2$ closest ENF proofs to E_i. Then we define the ENF score for v_i:

$$s(i) = \sum_{i \to j} \|E_i - E_j\|^2. \tag{1}$$

Eq. (1) calculates ENF scores $(s(1), ..., s(n))$ associated with validators v_1 to v_n respectively, and applies Krum rule to select the minimum ENF score as follows:

$$s^* = \underset{i \in \{1,...,n\}}{\operatorname{argmin}} (s(i)). \tag{2}$$

Finally, the PoENF puzzle problem is formally defined as:

Definition 4. *Proof-of-ENF - Given $G_i = \{E_1, E_2, ..., E_n\}$ collected by validator $v_i \in D$, the process of PoENF verifies whether a valid ENF proof E_j can meet the condition: $s(j) \leq s^*$. If yes, v_j wins the leader election and is qualified to propose a block; Otherwise, blocks generated by any v_j are rejected.*

Given the above definitions, the PoENF-enabled block generation procedures are presented in Algorithm 1. During the current block generation round slot sl_t, each validator $v_i \in D$ executes $generate\_block()$ function to probably propose a candidate block according to its collected ENF proofs in transactions pool. If the ENF score s_i is not greater than the target value s^*, then v_i can create a new block and broadcast it to the network. Otherwise, they are only allowed to verify blocks from other validators until current round finished. The closer s_i to s^*, the higher probability that v_i can propose a new block.

In block verification process, each validator calls $verify\_block()$ function to determine whether or not the received *new_block* can be accepted and appended to chain. The $Verify\_Sign()$ checks if a *new_block* sent by v_j has a valid signature σ_j, while $Verify\_TX()$ validates that all transactions recorded in *new_block* are sent from valid nodes and have the same Merkle tree root as *new_block.mt_root*. After validating that *new_block* is generated in the current round slot sl_t with correct chain header, a PoENF algorithm verifies if v_j has a valid ENF proof with minimum score. If all conditions are satisfied, the *new_block* is accepted into confirmed status, and v_i updates the head of local chain as $head(\mathcal{C}) = B_{i+1}$ accordingly. Otherwise, the *new_block* is rejected and discarded.

Algorithm 1 The PoENF-based block generation procedures.

procedure: generate_block(v_i)
  $hc \leftarrow \mathcal{H}(head(\mathcal{C}))$
  $height \leftarrow head(\mathcal{C}).height + 1$
  $mt\_root \leftarrow$ MTree($v_i.TX$)
  $[E_1, E_2, ..., E_n] \leftarrow$ ENF_vect($v_i.TX$)
  $enf\_score \leftarrow []$
  for E_i in $[E_1, E_2, ..., E_n]$ **do**
    $s_i \leftarrow \sum_{i \to j} \|E_i - E_j\|^2$
    enf_score.append(s_i)
  end for
  $s^* \leftarrow Min(enf\_score)$
  if $s_i \leq s^*$ **then**
    $new\_block \leftarrow (hc||mt\_root||v_i.TX||v_i.pk||sl.t||height)$
    $\sigma_i \leftarrow$ Sign($new\_block, v_i.sk$)
    return $(new\_block||\sigma_i)$
  end if
procedure: verify_block(new_block, σ_j)
  if Verify_Sign(new_block, σ_j) $\neq$ *True* OR
    Verify_TX(new_block) $\neq$ *True* **then**
    return *False*
  end if
  $hc \leftarrow \mathcal{H}(head(\mathcal{C}))$
  if $new\_block.height \neq head(\mathcal{C}).height + 1$ OR
    $new\_block.hc \neq hc$ **then**
    return *False*
  end if
  $[E_1, E_2, ..., E_n] \leftarrow$ ENF_vect($new\_block.tx\_list$)
  $enf\_score \leftarrow []$
  for E_i in $[E_1, E_2, ..., E_n]$ **do**
    $s_i \leftarrow \sum_{i \to j} \|E_i - E_j\|^2$
    enf_score.append(s_i)
  end for
  $s^* \leftarrow Min(enf\_score)$
  if $s_j > s^*$ **then**
    return *False*
  end if
  return *True*

4.3.3. Chain Extension Policies

In a PoENF block generation round, validators extend local chain based on a "*largest height of confirmed block*" rule, which requires that new blocks B_{i+1} are only appended to $head(\mathcal{C})$ by letting $B_{i+1}[pre\_hash] = \mathcal{H}(head(\mathcal{C}))$. The PoENF process allows that the probability of a block generated by validator v_i is related to the rank of its ENF proof E_i among the global ENF proof vectors G. However, G_i observed by validator v_i may vary due to network latency or misbehavior of Byzantine nodes. Thus, it is hard to guarantee that only one block is proposed during the each slot round, and the amount of candidate blocks could be between zero and the committee size K. Given the candidate blocks number $b \in [0, K]$, the chain extension rules are described as follows:

i) $b = 1$: if there is only one proposed candidate block B_{i+1}, then the block is accepted as confirmed status and updates the chain head as $head(\mathcal{C}) = B_{i+1}$.

ii) $b > 1$: if more than one candidate blocks are proposed, then all blocks are accepted as confirmed status. The $head(\mathcal{C})$ update follows two sub rules:

a) chain head points to a block that records most ENF proof transactions among other blocks; and
b) for the candidate blocks having the same size of ENF proof transactions, the block generated by validator who has the largest credit value becomes the chain head.

iii) $b = 0$: if none of the validators proposes a block at the end of current slot round, block generation follows a spin manner. As validators of current committee can be sorted by account address, we can calculate $ind = height \pmod K$. Thus, a validator at rank ind can also propose a candidate block in current round. The chain head update process follows the rule i) $b = 1$.

The rule i) covers a basic scenario to ensure that a new blocks is extended on chain head. The rule ii) handles the conflicting chain head update scenario when multiple validators propose valid blocks when they have different global ENF proof vectors G. It also discourages dishonest behaviors by using a smaller G to win block proposal right. The rule iii) guarantees the liveness in PoENF consensus process, such that at least one uniform block is generated to ensure chain extension even if a leader cannot propose a new block due to crash failures or attacks.

4.4. Voting-based Chain Finality Mechanism

Because of fork issues caused by network latency or deliberate attacks, the block proposal mechanism will inevitably produce multiple conflicting blocks, which are children blocks with the same parent block. Therefore, those proposed blocks in fact form an ever-growing *block tree* structure, as the upper part of Figure 3 shows. At the end of an epoch, the *head* with epoch height becomes a checkpoint that is used to resolve forks and finalize the chain history. Inspired by Casper [24] and Microchain [40], our EconLedger finality process adopts a voting-based finality mechanism overlaying the PoENF block generation to commit checkpoint block and finalize those already committed blocks on the main chain.

The chain finality protocol is mainly to identify a unique chain path on block tree by choosing a single child block from multiple children blocks with a common parent block. For efficiency purposes, the chain finality protocol is only executed on those *checkpoint* blocks rather than the entire block tree, and committee members vote for hashes of blocks instead of entire block contents. The chain finality ensures that only one path, including finalized blocks, becomes the main chain. Therefore, blocks generated in the new epoch are only extended on such a unique main chain.

4.5. Incentives and Punishment Strategies

Although the contributions of this manuscript are the performance improvement and security guarantees by EconLedger, this section briefly discusses incentives design while leaving detail analysis for future work. EconLedger uses an incentive mechanism to reward validators who behave honestly and make contributions in the PoENF block generation and chain finality process. At the end of a block generation cycle, transactions fees included in the confirmed block construct a rewarding fees pool that can be distributed to all validators in current round. The incentives mechanism uses ENF score to evaluate a validator's contribution, and rewarding fees that are distributed to $v_i \in D$ are proportional to its ENF score s_i. Let $S = \{s_1, s_2, ..., s_n\}$ denote ENF scores, the rewarding rule is defined as follow:

$$\gamma_i = \frac{\frac{1}{s_i}}{\sum_i^n \frac{1}{s_i}} \mathcal{R}, \tag{3}$$

where γ_i is rewarding fees that v_i obtains from the total rewarding fees $\mathcal{R}$ during current block generation round. The smaller ENF score of a validator, the higher rewarding fees it can gain. As the variations of ENF proofs from all honest nodes are trivial during ENF collection time, collected rewarding fees in $\mathcal{R}$ are almost evenly distributed to

Table 2: Configuration of Experimental Nodes.

Device	Dell Optiplex-7010	Dell Optiplex 760	Raspberry Pi 4 Model B
CPU	Intel Core TM i5-3470 (4 cores), 3.2GHz	Intel Core TM E8400 (2 cores), 3GHz	Broadcom ARM Cortex A72 (ARMv8) , 1.5GHz
Memory	8GB DDR3	4GB DDR3	4GB SDRAM
Storage	350G HHD	250G HHD	64GB (microSD)
OS	Ubuntu 16.04	Ubuntu 16.04	Raspbian (Jessie)

honest contributors. As ENF fluctuations are random generated from power grid and varying at different time, thus, Byzantine nodes can only gain marginal benefits by using duplicated or arbitrary ENF proofs that have large ENF scores.

In addition to rewarding fees, the credit stake c_i of a honest validator v_i will also increase by one as a reputation reward. The higher credit stake c, the higher probability that a validator is selected as a PoENF committee member. Unlike PoS, credits in EconLedger are not directly associated with any type of currency, and they are not transferable in any format of transactions. Therefore, all users are encouraged to behave honestly to gain more benefits by increasing their reputation credits. Moreover, credit stake c of a node cannot excel a upper-bounded limitation C_{max}, for instance, no more than 10. Therefore, an adversary cannot simply accumulate its credit stake to achieve mining centralization and then control the majority power of the network.

A punishment strategy is also designed to discourage dishonest behaviors, like withholding its ENF proof, proposing multiple blocks in current round or violating chain extension rules. After PoENF committee selection, each $v_i \in D$ must deposit a fix amount of fees to its *security stake* sc_i. If any misbehaving actions in consensus process v_i are detected, the balance of sc_i will be slashed as punishment. In addition, its credit stake c_i also decreases by one. Given assumption that an adversary can only compromise no more than f nodes on the network, slashing security deposit rule can increase the financial cost if attackers use these compromised nodes to disturb consensus protocol. While reducing credit stake leads to the lower probability that Byzantine nodes can be selected as committee members.

5. Experiment and Evaluation

To verify the proposed EconLedger, a concept-proof prototype is implemented in Python, which consists of approximate 3100 lines of code. We adopt Flask [41], which is a light micro-framework for Python application, to implement networking and web service APIs for EconLedger node. All cryptographic functions are developed on the foundation of standard python lib: cryptography [42], like using RSA for key generation and digital signature, and using SHA-256 for all hash operations. As a lightweight and embedded SQL database engine, SQLite[43] is adopted to manage on-chain storage, such as ledger data and peering nodes information.

5.1. Experimental Setup

Table 2 describes the devices used for the experimental study. The prototype is deployed on a small-scale local area network (LAN) that consists of multiple desktops and IoT devices. The prototype of EconLedger emulates an office building setting: a Dell Optiplex-7010 functions as a monitor server to collect data from scattered IoVT services deployed at different locations of the building, while all Raspberry Pi (RPi) boards play the role of edge devices that process raw video streams from separate cameras. All device can work as validators and perform the PoENF consensus protocol. Dell Optiplex 760

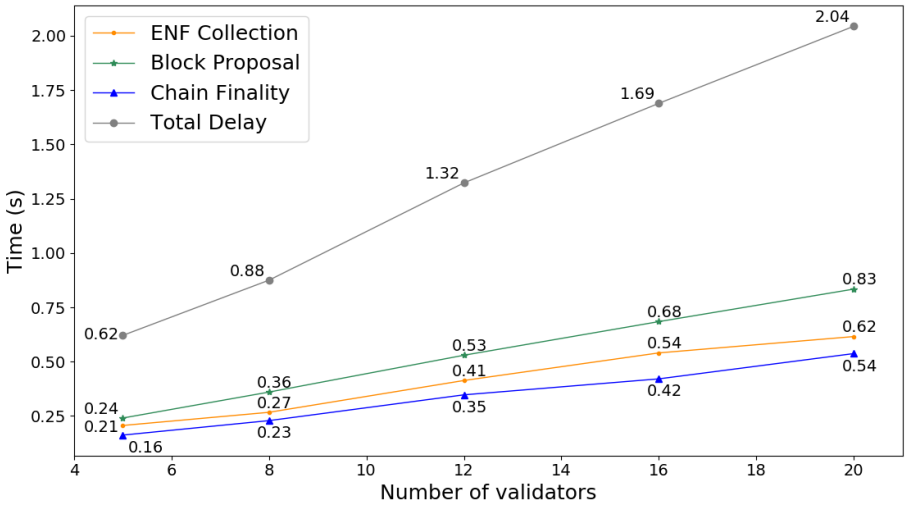

Figure 4. Latency for an epoch cycle of PoENF consensus with different committee size.

desktop functions as edge server, and five desktops are configured as sites in our private Swarm network. To make comparative evaluation between EconLedger and existing blockchain benchmarks, test cases are also conduct on Ethereum [44] and Tendermint [45] networks. In our private Ethereum network setup, six miners are deployed on six separate desktops. Tendermint runs on a test network with 20 validators, and each validator is hosted on a RPi device.

5.2. Performance Evaluation

To evaluate the performance of the running EconLedger under an IoVT-based edge network environment, a set of experiments is conducted by executing multiple complete epoch cycles of PoENF consensus protocol within a dynasty. The computation cost by message encryption and decryption are not considered during the test. As Krum in PoENF requires $n \geq 2f + 3$ and voting-based chain finality depends on a majority condition that needs $n \geq 3f + 1$, the minimum PoENF committee size is with five members, such that any $K \geq 5$ can meet both security requirements. Given 60 seconds per sliding window used in ENF fluctuations extraction, we let the ENF proof vector size d=60. We conducted 100 Monte Carlo test runs for each test scenario and used the average of results for evaluation.

5.2.1. Network Latency

Figure 4 presents the network latency for EconLedger to complete an entire epoch round of PoENF consensus protocol given the number of validators varying from 5 to 20. For each test point, we let all validators perform task simultaneously and wait until the bundle of tasks is finished. The latency includes the round trip time (RTT) and service processing time on the remote host. As broadcasting an ENF tx needs $\mathcal{O}(K)$ communication complexity and $K \times \mathcal{O}(1)$ computation complexity for verification. Thus, the total complexity is $\mathcal{O}(K)$, such that latency of ENF collection $\mathcal{T}_{ec}$ is linear scale to committee size K. Chain finality requires all validators to broadcast their vote among committee members, so that it has the same complexity as ENF collection. Thus, the delay of chain finality $\mathcal{T}_{cf}$ is almost linear scale to K.

The green line in Figure 4 shows the latency of block proposal $\mathcal{T}_{bp}$, which indicates how long a proposed block could be accepted by all validators in PoENF committee. The communication complexity of block proposal is $\mathcal{O}(K)$, which is similar to ENF collection and chain finality. However, during block generation and verification processes, PoENF algorithm requires a validator use Eq.1 to compute ENF scores based on collected proofs from others, and it has computation complexity of $\mathcal{O}(K^2 d)$. As a result, the total complexity of block proposal is $\mathcal{O}(K^2 d + K)$. In general, ENF samples size d is a small

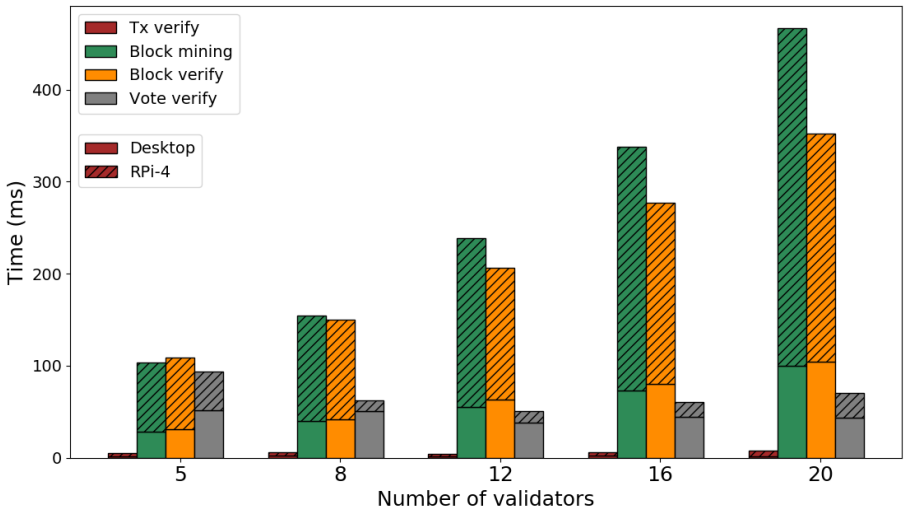

Figure 5. Computation overhead for stages of running PoENF consensus on host. Comparative evaluations on platform benchmark.

value like 60, and it has less effect on computation cost than K does. Thus, $\mathcal{T}_{bp}$ is almost linear scale to the $\mathcal{O}(K^2)$. The total latency shows that EconLedger takes about 2 seconds to finish an epoch cycle of PoENF committee consensus $(\mathcal{T}_{ec} + \mathcal{T}_{bp} + \mathcal{T}_{cf})$ given the committee size $K = 20$.

5.2.2. Computation Overhead

Figure 5 shows service processing time of key procedures in PoENF consensus given different platform benchmarks. As verifying a tx or vote only involves $\mathcal{O}(1)$ computation complexity, the service processing time is almost stable (about 2 ms for tx verification and 50 ms for vote verification) on all benchmarks. Compared to tx verification that simply checks the validity of a tx then buffer it into system memory, vote verification involves more computation on resolving forks and database operations to store valid votes. Thus, vote verification incurs more latency than tx verification does. As the most computing intensive stages, both block mining and verify rely on procedures in PoENF consensus algorithm with the computation complexity of $\mathcal{O}(K^2 d)$. Therefore, the computation cost on all devices dramatically increases as scaling up K. Given different computation capacity of benchmarks, RPi-4 needs 2.5× processing time that Desktop dose.

5.2.3. Data Throughput

To evaluate overhead of running EconLedger on communication channel, we consider volumes of message propagation and data throughput during key steps of PoENF consensus protocol. Figure 6 demonstrates data transmission for different stages of PoENF consensus with varying committee size. In our EconLedger prototype, each ENF transaction has fixed size d_{tx}=430 Bytes, and a vote has fixed size d_{vt}=589 Bytes. Given total communication complexity of $\mathcal{O}(K^2)$ in both ENF collection and chain finality, data transmission of ENF collection $\mathcal{D}_{ec}$=$d_{tx} \times K^2$, and data transmission of chain finality $\mathcal{D}_{cf}$=$d_{vt} \times K^2$. Thus, communication overheads incurred by ENF collection and chain finality are linear scale to K^2.

Each block has a fixed header d_{head}=613 Bytes along with a transactions list with size d_{txs}=$d_{tx} \times K$, and we can get block size d_B=$d_{head} + d_{txs}K$. Therefore, the block size d_B is linear scale to K. Assuming an ideal case that only one valid block is proposed during each epoch cycle, data transmission of block proposal $\mathcal{D}_{bp}$=$d_B \times K$=$d_{head}K + d_{txs}K^2$, such that communication overhead is almost scale to K^2. But for the worst case that every

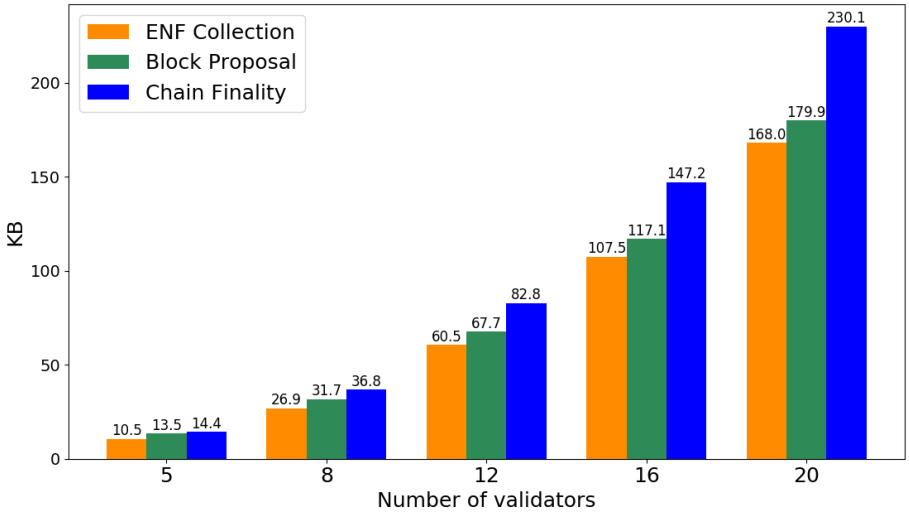

Figure 6. Communication overhead of running an epoch round of PoENF consensus. Comparative evaluations on different committee size.

Table 3: Data Throughputs vs. Committee Sizes.

Committee Size	5	8	12	16	20
Block Size (KB)	2.7	3.9	5.6	7.3	8.9
Throughput (KB/s)	61.9	108.3	159.8	220.1	283.3

validator proposed a candidate block such that $\mathcal{D}_{bp}$=$d_B \times K^2$, it can introduce huge communication cost as scaling up K.

The data throughput could be specified as $Th = \frac{\mathcal{D}_{ec}+\mathcal{D}_{bp}+\mathcal{D}_{cf}}{\mathcal{T}_{ec}+\mathcal{T}_{bp}+\mathcal{T}_{cf}}$ (KB/s), where KB/s means KBytes per second. With variant committee sizes, corresponding block size and data throughput are calculated as shown in Table 3. Given a fixed ENF transaction size, increasing the committee size allows committing more ENF proofs, and therefore reach a higher data throughput at the cost of latency. In the test case of $K = 20$, EconLedger implies a theoretical maximum data rate of 283 KB/s, which can meet bandwidth conditions in majority of LAN-based IoVT systems.

5.3. Comparative Evaluation

As a key parameter in blockchain network, transaction tx committed time indicates how long a tx can be finalized in a new block on distributed ledger, and it is closely related to block confirmation time in consensus protocol. Given different blockchain benchmarks, we evaluate the end-to-end time latency of committing transactions along with other key performance metrics. For Ethereum, we use smart contract to record transactions on blockchain. For the Tendermint network, we use built-in kvstore as an ABCI (Application BlockChain Interface) app to save transactions.

We conducted 50 Monte Carlo test runs, which a node sends a 1KB tx per second (TPS) to blockchain network and waits until tx has been confirmed on distributed ledger. Figure 7 shows the distributions of time delay for committing transactions given different blockchain networks. Each green bar indicates standard deviation with a mean represented by red dot. The gray line shows whole data range and black star is median. Tendermint uses a BFT consensus protocol to achieve high efficiency, therefore, mean of tx committed time is about 3 s given 1 voting round per second. Unlike Tendermint, Ethereum relies on a probabilistic PoW consensus, which has variable block confirmation time. Thus, tx committed time in Ethereum network is varying with largest standard deviation. To guarantee synchronous epoch rounds for PoENF consensus, we set $\mathcal{T}_\Delta$

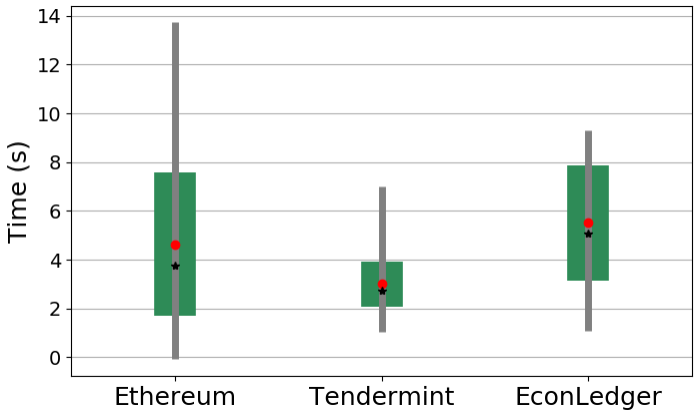

Figure 7. Time latency for committing transactions. Comparative evaluations on different blockchain networks.

Table 4: Comparative evaluation of launching transactions on different blockchain platforms.

	Ethereum	Tendermint	EconLedger
tx **committed time (s)**	4.6	3.0	5.5
tx **rate (tx/s)**	124	334	436
CPU usage (%)	103	38.6	15.2
Memory usage (MB)	1,200	72	80

conservatively to 2 s based on maximum time to ensure txs and blocks propagation in a P2P consensus network including 20 validators. Hence, the range of latency in EconLedger is smaller than Ethereum, and tx committed time is almost stable (about 5.5 s).

Table 4 provides a comprehensive performance of committing transactions on different blockchain networks regarding several key performance matrices. Given above tx committed time, which uses mean in Figure 7, tx rate tx/s is evaluate by calculating how many tx can be processed per second in blockchain network. Ethereum block size is bounded by how many units of gas can be spent per block, which is known as the block gas limit [46]. Currently the maximum block size is around 12,000,000 Gas (Accessed at July 20, 2020) and the base cost of any transaction is about 21,000, thus, each block in Ethereum can include about 571 transactions. In our private Ethereum network, we can get tx rate is (571.4/4.6)≈124 tx/s. Tendermint and EconLedger both use fixed 1MB block. Given 1 KB per transaction, a block in Tendermint can store the maximum of 1000 transactions, thus, tx rate is (1000/2.9)≈344 tx/s. For EconLedger, each transaction is about 430 Bytes such that a block can record the maximum of 2400 transactions, then it achieves higher tx rate (2400/5.5)≈436 tx/s.

To evaluate resource consumption by running blockchain benchmarks, we use "top" command to monitor system performance of machines. We consider CPU and memory usage on desktop (Ethereum miner) and Rpi (Tendermint and EconLedger validator). Due to computation intensive PoW algorithm, the mining process of Ethereum almost occupies full CPU capacity and consumes about 1.2GB memory. Therefore, such a huge computation cost prevents resource constrained edge devices mining in Ethereum network. Unlike Ethereum, Tendermint and EconLedger use lightweight consensus algorithms to achieve efficiency in CPU and memory usage, such that they are both suitable for deploying validators on edge devices. EconLedger almost has the same amount of memory usage as Tendermint in running time. However, EconLedger has the higher tx committed time than Tendermint does, and it only needs 40% computation resource that Tendermint does.

5.4. Performance and Security Analysis

5.4.1. Performance Improvements

Given above numerical results in terms of processing time and running time resource usages, our PoENF consensus is more computation efficient than those PoW-based methods. Such a lightweight property of PoENF is promising to reduce energy consumption on mining process and lower demands on system capability for participants. Thus, resource-limited IoVT devices can directly work as validators (miners) rather than depend on supports from an intermediate consensus layer by outsourcing mining tasks on fog networks or cloud servers. Compared with these hardware dependent solutions, like REM based on Intel SGX and PoR requiring on large local storage, our PoENF consensus relies on a platform independent algorithm to extract ENF-containing multimedia signals from recordings as ENF proofs. Therefore, it is promising to address heterogeneity issues as we integrate blockchain technology with IoVT systems that include multiple non-standard platforms.

EconLedger achieves communication efficiency by executing consensus protocol within a random selected PoENF committee. Such a small scale consensus network imposes low level of data transfer overhead on IoVT systems at the network of edge which has limited bandwidth. In addition, communication complexity for each validator is linear scale to PoENF committee size shown in Figure 6. Thus, limited data transmission also means lower energy consumption on device during communication handling tasks. Unlike non-scalable BFT-based solutions which relies on a pre-fixed set of validators, EconLedger aims to improve scalability by requiring a random elected consensus committee to delegate other nodes of network. As a trade-off, EconLedger is actually a partially decentralized blockchain network.

In EconLedger, raw data are saved into off-chain storage deployed on a DDB network, while only references of data are encapsulated into transactions that are finalized on distributed ledger (on-chain storage). As reference is a fixed length of hash value disregarding format or size of source data, such light transactions can be used to verified complicated data in use over IoVT systems, like multimedia recordings, contextual information and trained models, ect.. Moreover, each tx has fixed and small size, such that a block can recorded more txs. As a result, txs rate is increased given that block confirmation time is stable.

5.4.2. Committee Randomness Security

We assume that an adversary has limited capacity such that he/she is subject to the usual cryptographic hardness assumptions, and honest nodes never share their keys with each others or disclose the input string x of the VRF function before the end of randomness generation. Therefore, members of a new committee could be completely random owning to unpredictability property of the VRF-based randomness string generation. In addition, given assumption that an adversary could only control up to f byzantine validators, the chain finality achieves safety by making agreements on checkpoints if current PoENF committee has no less than $2f + 1$ honest members. Therefore, the adversary has at most $m = 1/4$ chance per round to control the checkpoint voting process. As a result, the probability that an adversary controls n consecutive checkpoint is upper-bounded by $P[X \geq n] = \frac{1}{4^n} < 10^{-\lambda}$. For $\lambda = 6$, the adversary will control at most ten consecutive chain finality runs.

5.4.3. PoENF Consensus Security

Unlike PoW and PoS consensus protocols that are vulnerable to mining centralization, whether a validator can become winner in current PoENF block proposal round depends on its ENF score rather than its controlled computation power or cryptocurrency stakes. Thus, an adversary cannot control mining process by solo increasing investments on computation resource or owned coins. Moreover, Krum rule adopted in ENF score calculation chooses $n - f - 2$ closet ENF proofs and precludes those $f - 1$ Byzantine

proofs that are far away. Thus, all honest validators can output the same minimum ENF score as long as $n \geq 2f + 3$, and our PoENF can prevent against ENF proof positioning attacks.

In PoENF consensus, all honest validators only accepts valid blocks generated in current epoch round, thus, *correctness (validity)* is ensured. In addition, PoENF achieves *consistency (agreement)* by requiring all honest validators to update their local chain head according to chain extension policies. At the end of a PoENF block proposal round, every honest validator should either accept valid transactions that are saved into a confirmed block as local chain header, or reject all transactions by extending a empty block on local chain header. Such a *liveness (termination)* property ensures that all valid ENF transactions are processed within the block generation round. Furthermore, voting-based chain finality can guarantee *safety* which requires all honest validators form a same total order of finalized blocks appended on the global unique main chain.

5.4.4. Analysis of Possible attacks

1. Double spending attacks: In a double spending scenario, an adversary attempts to revert a transaction which has been finalized on distributed ledger. The chain finality mechanism ensures total order and persistence of data recorded on ledger. Thus, once a transaction is finalized in checkpoint block, all other honest nodes will work on finalized chain and disregard the double spending transactions from attackers.

2. Free-riding attacks: There is a possibility of free-riding attacks that some lazy nodes only benefit from the security service without fulfilling their responsibilities in EconLedger network, like not forward messages or submit ENF proofs. The punishment strategies can prevent against free-riding attacks by reducing credit stake of dishonest nodes or even isolating them from the whole network.

3. Selfish-mining attacks: In a selfish-mining attack, the adversary tries to withhold blocks and release them strategically to reduce chain growth and increase the relative ratio of his proposed blocks. In PoENF consensus, only valid blocks generated in current round can be accepted by honest validators, while those outdated blocks are discarded. Moreover, withholding blocks is a type of misbehavior in PoENF and it decreases both profits and credit of a dishonest node. Therefore, selfish-mining is unprofitable for rational validators according to reward and punishment strategies.

4. ENF-proof replay attacks: The adversary can launch replay attack by sending duplicated ENF proofs. As ENF fluctuations of power grid vary as time changes such that those duplicate ENF proofs generally output large ENF scores. As a result, these Byzantine validators have marginal chances to propose valid blocks. Furthermore, ENF-proof replay attack can be detected by analyzing ENF proofs on EconLedger. Thus, identifying misbehavior and isolating suspicious nodes can improve system robustness. While we leave ENF-based detection topics to future work.

6. Conclusions

This paper presents EconLedger, a lightweight and secure-by-design distributed ledger to enhance trust and security properties for smart IoVT systems at the edge. The EconLedger combines an efficient PoENF consensus mechanism with a deterministic voting-based chain finality to achieve safety and liveness. Using on-chain ledger and DDB enabled off-chain storage, EconLedger reduces storage overheads on validators and guarantees security and resilience of data sharing in distributed IoVT network. Experimental results based on a prototype demonstrate that it achieves higher computation efficiency and tx throughput than benchmarks.

Our on-going efforts include validating the proposed architecture in a real-world video streams context, simulating attack scenarios, and ensuring overall system-design stability, efficiency and effectiveness. This includes a fully functional prototype implementation and a comprehensive performance analysis and assessment on security

properties. Moreover, We will also design an ENF-based visual layer attack detection and prevention system atop of EconLedger fabric.

Acknowledgement

This work is supported by the U.S. National Science Foundation (NSF) via grant CNS-2039342 and the U.S. Air Force Office of Scientific Research (AFOSR) Dynamic Data and Information Processing Program (DDIP) via grant FA9550-21-1-0229. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U. S. Air Force.

Author Contributions: Conceptualization, R.X., D.N and Y.C.; Methodology, R.X. and D.N.; Software, R.X. and D.N.; Validation, R.X., D.N. and Y.C.; Formal analysis, R.X., D.N. and Y.C.; Investigation, R.X.; Resources, R.X. and D.N.; Data Curation, R.X.; Writing–Original Draft Preparation, R.X. and D.N.; Writing–Review and Editing, R.X. and Y.C.; Visualization, R.X.; Supervision, Y.C.; Project Administration, Y.C. All authors have read and agreed to the published version of the manuscript.

Abbreviations

Abbreviations

The following abbreviations are used in this manuscript:

ABCI	Application BlockChain Interface
BFT	Byzantine Fault Tolerant
CCD	Charge-coupled Device
COMS	Complementary Metal Oxide Semiconductor
CPU	Central Processing Unit
DAG	Directed Acyclic Graph
DDB	Decentralized Database
DHT	Distributed Hash Table
DLT	Distributed Ledger Technology
DPA	Distributed Pre-image Archive
ENF	Electrical Network Frequency
IoVT	Internet of Video Things
LAN	Local Area Network
M2M	Machine-to-Machine
P2P	Peer-to-Peer
PBFT	Practical Byzantine Fault Tolerant
PKI	Public Key Infrastructure
PoENF	Proof-of-ENF
PoET	Proof-of-Elapsed Time
PoR	Proofs-of-Retrievability
PoS	Proof-of-Stake
PoUW	Proof-of-Useful-Work
PoW	Proof-of-Work
PoX	Proof-of-X-concept
PVSS	Publicly Verifiable Secret Sharing
REM	Resource Efficient Mining
RSA	Rivest–Shamir–Adleman
RTT	Round Trip Time
RPi	Raspberry Pi
SGX	Software Guard Extensions
SMR	State Machine Replication
TPS	Transactions per Second
VRF	Verifiable Random Function

References

1. Xu, R.; Nikouei, S.Y.; Chen, Y.; Polunchenko, A.; Song, S.; Deng, C.; Faughnan, T.R. Real-time human objects tracking for smart surveillance at the edge. 2018 IEEE International Conference on Communications (ICC). IEEE, 2018, pp. 1–6.
2. Nikouei, S.Y.; Xu, R.; Nagothu, D.; Chen, Y.; Aved, A.; Blasch, E. Real-time index authentication for event-oriented surveillance video query using blockchain. 2018 IEEE International Smart Cities Conference (ISC2). IEEE, 2018, pp. 1–8.
3. Nikouei, S.Y.; Chen, Y.; Aved, A.; Blasch, E.; Faughnan, T.R. I-safe: Instant suspicious activity identification at the edge using fuzzy decision making. Proceedings of the 4th ACM/IEEE Symposium on Edge Computing, 2019, pp. 101–112.
4. Ali, M.S.; Vecchio, M.; Pincheira, M.; Dolui, K.; Antonelli, F.; Rehmani, M.H. Applications of blockchains in the Internet of Things: A comprehensive survey. *IEEE Communications Surveys & Tutorials* **2018**, *21*, 1676–1717.
5. Nikouei, S.Y.; Chen, Y.; Faughnan, T.R. Smart surveillance as an edge service for real-time human detection and tracking. 2018 IEEE/ACM Symposium on Edge Computing (SEC). IEEE, 2018, pp. 336–337.
6. Zhou, Q.; Huang, H.; Zheng, Z.; Bian, J. Solutions to scalability of blockchain: A survey. *IEEE Access* **2020**, *8*, 16440–16455.
7. Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.
8. Miller, A.; Juels, A.; Shi, E.; Parno, B.; Katz, J. Permacoin: Repurposing bitcoin work for data preservation. 2014 IEEE Symposium on Security and Privacy. IEEE, 2014, pp. 475–490.
9. Castro, M.; Liskov, B.; others. Practical Byzantine fault tolerance. OSDI, 1999, Vol. 99, pp. 173–186.
10. Hajj-Ahmad, A.; Garg, R.; Wu, M. ENF-based region-of-recording identification for media signals. *IEEE Transactions on Information Forensics and Security* **2015**, *10*, 1125–1136.
11. Nagothu, D.; Chen, Y.; Blasch, E.; Aved, A.; Zhu, S. Detecting malicious false frame injection attacks on surveillance systems at the edge using electrical network frequency signals. *Sensors* **2019**, *19*, 2424.
12. Swarm. https://ethersphere.github.io/swarm-home/. accessed: Jan. 2, 2020.
13. Bollen, M.H.; Gu, I.Y. *Signal processing of power quality disturbances*; Vol. 30, John Wiley & Sons, 2006.
14. Grigoras, C. Applications of ENF analysis in forensic authentication of digital audio and video recordings. *Journal of the Audio Engineering Society* **2009**, *57*, 643–661.
15. Hajj-Ahmad, A.; Garg, R.; Wu, M. Instantaneous frequency estimation and localization for ENF signals. Proceedings of The 2012 Asia Pacific Signal and Information Processing Association Annual Summit and Conference. IEEE, 2012, pp. 1–10.
16. Nagothu, D.; Chen, Y.; Aved, A.; Blasch, E. Authenticating Video Feeds using Electric Network Frequency Estimation at the Edge. *EAI Endorsed Transactions on Security and Safety* **2021**.
17. Buterin, V.; others. A next-generation smart contract and decentralized application platform. *white paper* **2014**.
18. Juels, A.; Kaliski Jr, B.S. PORs: Proofs of retrievability for large files. Proceedings of the 14th ACM conference on Computer and communications security, 2007, pp. 584–597.
19. Zhang, F.; Eyal, I.; Escriva, R.; Juels, A.; Van Renesse, R. {REM}: Resource-Efficient Mining for Blockchains. 26th {USENIX} Security Symposium ({USENIX} Security 17), 2017, pp. 1427–1444.
20. Sawtooth Documentation. https://sawtooth.hyperledger.org/docs/core/releases/latest/introduction.html. accessed: Jan. 2, 2020.
21. King, S.; Nadal, S. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *self-published paper, August* **2012**, *19*.
22. Lamport, L.; Shostak, R.; Pease, M. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)* **1982**, *4*, 382–401.
23. Hyperledger Fabric. http://https://github.com/hyperledger/fabric. accessed: Jan. 2, 2020.
24. Buterin, V.; Griffith, V. Casper the friendly finality gadget. *arXiv preprint arXiv:1710.09437* **2017**.
25. Bao, Z.; Shi, W.; He, D.; Chood, K.K.R. IoTChain: A three-tier blockchain-based IoT security architecture. *arXiv preprint arXiv:1806.02008* **2018**.
26. Tuli, S.; Mahmud, R.; Tuli, S.; Buyya, R. Fogbus: A blockchain-based lightweight framework for edge and fog computing. *Journal of Systems and Software* **2019**, *154*, 22–36.
27. Sagirlar, G.; Carminati, B.; Ferrari, E.; Sheehan, J.D.; Ragnoli, E. Hybrid-iot: Hybrid blockchain architecture for internet of things-pow sub-blockchains. 2018 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), 2018, pp. 1007–1016.
28. IOTA Foundation. IOTA Data Marketplace. https://data.iota.org. accessed: Jan. 2, 2020.
29. Popov, S. The tangle. *cit. on* **2016**, p. 131.
30. Xu., R.; Chen, Y.; Blasch, E.; Chen, G. Exploration of blockchain-enabled decentralized capability-based access control strategy for space situation awareness. *Optical Engineering* **2019**, *58*, 041609.
31. Xu, R.; Chen, Y.; Blasch, E.; Chen, G. Blendcac: A smart contract enabled decentralized capability-based access control mechanism for the iot. *Computers* **2018**, *7*, 39.
32. Nagothu, D.; Schwell, J.; Chen, Y.; Blasch, E.; Zhu, S. A study on smart online frame forging attacks against video surveillance system. Sensors and Systems for Space Applications XII. International Society for Optics and Photonics, 2019, Vol. 11017, p. 110170L.
33. Xu, R.; Nagothu, D.; Chen, Y. Decentralized video input authentication as an edge service for smart cities. *IEEE Consumer Electronics Magazine* **2021**.

34. Lamport, L. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM* **1978**, *21*, 558–565.
35. Swarm Docs. https://swarm-guide.readthedocs.io/en/latest/introduction.html. accessed: Jan. 2, 2020.
36. Maymounkov, P.; Mazieres, D. Kademlia: A peer-to-peer information system based on the xor metric. International Workshop on Peer-to-Peer Systems. Springer, 2002, pp. 53–65.
37. Gilad, Y.; Hemo, R.; Micali, S.; Vlachos, G.; Zeldovich, N. Algorand: Scaling byzantine agreements for cryptocurrencies. Proceedings of the 26th Symposium on Operating Systems Principles. ACM, 2017, pp. 51–68.
38. Stadler, M. Publicly verifiable secret sharing. International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 1996, pp. 190–199.
39. Blanchard, P.; Guerraoui, R.; Stainer, J.; others. Machine learning with adversaries: Byzantine tolerant gradient descent. Advances in Neural Information Processing Systems, 2017, pp. 119–129.
40. Xu, R.; Chen, Y.; Blasch, E. Microchain: A Light Hierarchical Consensus Protocol for IoT Systems. In *Blockchain Applications in IoT Ecosystem*; Springer, 2021; pp. 129–149.
41. Flask: A Pyhon Microframework. http://flask.pocoo.org/. accessed: Jan. 2, 2020.
42. pyca/cryptography documentation. http://pyca/cryptography. accessed: Jan. 2, 2020.
43. SQLite. https://www.sqlite.org/index.html. accessed: Jan. 2, 2020.
44. Ethereum Homestead Documentation. http://www.ethdocs.org/en/latest/index.html. accessed: Jan. 2, 2020.
45. Kwon, J. Tendermint: Consensus without mining. *Draft v. 0.6, fall* **2014**, *1*, 11.
46. What's the Maximum Ethereum Block Size? https://ethgasstation.info/blog/ethereum-block-size/. accessed: Jan. 2, 2020.