future internet

MDPI

Article

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

Ronghua Xu, Deeraj Nagothu and Yu Chen *

Department 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). The robustness and security of IoVT systems are essential for a sustainable urban environment. While blockchain technology has shown great potential in enabling trust-free and decentralized security mechanisms, directly embedding cryptocurrency oriented blockchain schemes into resource-constrained Internet of Video Things (IoVT) networks at the edge is not feasible. By 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. The decentralized database (DDB) is adopted in order 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)

Citation: Xu, R.; Nagothu, D.; Chen, Y. EconLedger: A Proof-of-ENF Consensus Based Lightweight Distributed Ledger for IoVT Networks. *Future Internet* **2021**, *13*, 248. https://doi.org/10.3390/fi13100248

Academic Editors: Ahad ZareRavasan, Taha Mansouri, Michal Krčál and Saeed Rouhani

Received: 7 September 2021
Accepted: 22 September 2021
Published: 24 September 2021

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 situational 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 stream processing is becoming 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 for accommodating 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, such as situation contextual features, are shared among participants using 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 attention for its potential to revolutionize multiple areas of the economy and

society. The inherent security guarantees of blockchain lay down the foundations of serverless record keeping, without the need for centralizing trusted third-party authorities [4]. Blockchain runs on a decentralized peer-to-peer (P2P) network in order to securely store and verify data without relying on a centralized trust authority. The decentralization removes the risk of singular point of failures and mitigates bottleneck performances, 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 renders IoVT systems more efficient, scalable, and secure. However, directly integrating cryptocurrency-oriented blockchains into resource constrained IoVT systems is difficult in terms of handling 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, such as Proof-of-Work (PoW) [7], Proofs-of-Retrievability (PORs) [8], or Practical Byzantine Fault Tolerant (PBFT) [9], which come 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.

The Electrical Network Frequency (ENF) is the power supply frequency which fluctuates around its nominal frequency (50/60 Hz). The frequency fluctuations vary based on 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 require 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 that presents a valid ENF proof with minimal squared-distance-based score is qualified to generate a new block. Thus, the PoENF consensus mechanism not only achieves efficiency without high demand of mining resource or hardware platform support but 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], which is 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 untrusted devices under a distributed network environment. Moreover, a network with permission 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 detailed explanation of the key components and work flows;
(2) A novel PoENF consensus mechanism is proposed, which improves resource efficiency and achieves a higher throughput than PoW-based blockchains;

(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;
(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 with respect 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

This section introduces how ENF can be generated from multimedia streams and how ENF can be used for the environmental fingerprint. Following that, we describe typical consensus protocols in blockchain and provide related work on IoT-blockchain integration.

2.1. ENF as a Region-of-Recording Fingerprint

ENF is the supply frequency in power distribution grids, which has a nominal frequency of 50 Hz or 60 Hz 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 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 for identifying 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 (120 Hz). 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 semiconductors (CMOSs) and charge-coupled device (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]. The estimation of ENF involves various signal processing techniques such as power spectral analysis and spectrogram-based techniques, which are beyond the scope of this paper.

2.2. Consensus Protocols for Blockchain

This section introduces consensus protocols regarding diverse blockchain networks that are typically classified into permissionless blockchain (e.g., Nakamoto protocol) or permissioned blockchain (e.g., PBFT).

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 such as Ethereum [17]. The Nakamoto protocol adopts a computation-intensive PoW, which requires all participants to compete for rewards through a cryptographic block-hash value

discovery racing game. The PoW consensus demonstrates 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 in terms of computation and energy consumption such that it is not affordable on resource-constrained IoT devices.

In order to improve 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 distributed storage of archival data. The Permacoin adopts PORs [18], which require miners to present random access to a copy of a file from local storage as valid proof for successfully minting money. Permacoin requires participants to invest in 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 models inherent in Intel Software Guard Extensions (SGX). The REM utilizes a Proof-of-Useful-Work (PoUW) consensus protocol, which requires miners to provide trustworthy measurements 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 the stale chip problem but also yields the smallest amount of mining waste.

In order 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 to only consume limited computational resources in order to generate new blocks. Similarly 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, such as 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 the 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, similarly to PoW and PoS. In Casper, a fixed set of validators executes 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 in order 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, the IoTChain [25] proposes a three-tier blockchain-based IoT architecture, which allows regional nodes to perform any lightweight consensus, such as PoS and PBFT. IoTChain only provides simulation results on communication cost of transactions; however, key metrics in the consensus layer, such as 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 hybrid blockchain-IoT architecture in order to improve scalability and interoperability among sub-blockchains. In HybridIoT, a BFT inter-connector framework functions as a global consortium blockchain to link multiple PoW sub-blockchains. However, using PoW consensus in sub-blockchain networks still imports computation and storage overhead on IoT devices if they are deployed as full nodes. IoTA [28] aims to enable 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 the 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 an 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 DDB technology to enable trusted off-chain storage, which reduces storage overhead caused by directly storing raw data on the public distributed ledger.

3. EconLedger: Rationale and Architecture

This section provides a comprehensive overview of EconLedger system architecture consisting of the following: (1) upper-level IoVT application layer; and (2) Econledger fabric enabled security networking infrastructure. Following that, we explain the network model of EconLedger with basic security assumptions and describe an efficient hybrid on-chain and off-chain storage structure based on the DDB system.

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 a user. Figure 1 is the system architecture of EconLedger.

3.1.1. IoVT Application

The upper-level IoVT application utilizes an EconLedger fabric to enable decentralized video analytic services 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 such as 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, such as object detection and situational contextual features extraction. Thus, cameras associated 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 the information visualization unit, which provides video recordings and smart applications for authorized users.

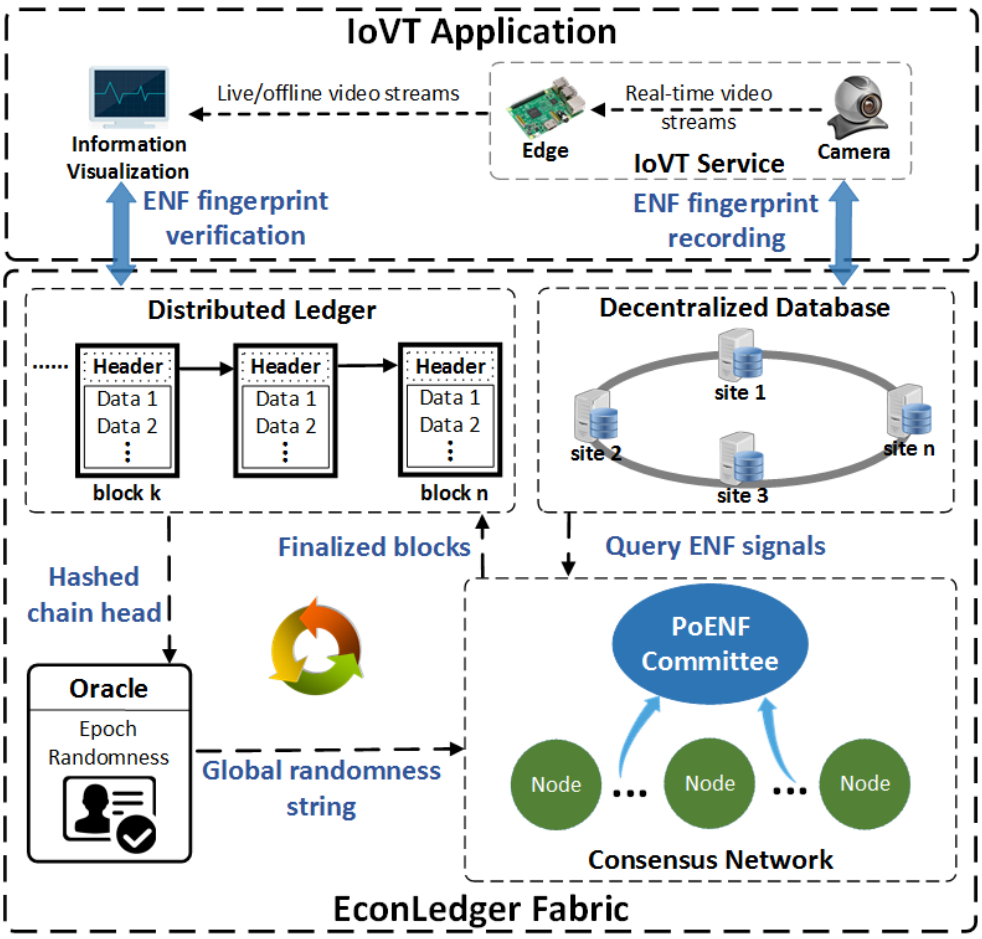

Figure 1. The EconLedger system architecture.

To prevent visual layer attacks, IoVT service extracts ENF signals from video streams as an environmental fingerprint, which is stored into DDB and 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 solid ground truth for video authenticity verification. By calculating correlation coefficients among ENF signals extracted from video recordings with an agreed ENF estimate recorded on distributed ledger, the 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 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 the devices launch transactions that include hashed references of raw data along with valid signatures. As transactions store fixed-length 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 message 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, scalability is improved at the cost of partial decentralization by a PoENF consensus committee.

Meanwhile, a random PoENF committee rotation strategy ensures that 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 decreases exponentially even if the current PoENF committee is compromised.

3.2. Network Model

EconLedger relies on a permissioned network, and we assume that the system administrator is a trust oracle for maintaining global identity profiles for all valid nodes. We adopt a standard asymmetrical algorithm such as Rivest–Shamir–Adleman (RSA) for key generation ($RSA.gen$) and digital signature scheme ($RSA.sign$, $RSA.verify$). During the registration process, signing-verification key 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 the 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 the 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 $\mathcal{T}_\Delta$. The time window of sl_t should be sufficient to guarantee that the message transmitted by a sender is received by its intended recipients (accounting for local time discrepancies and network delays). Thus, we require $sl_t \geq \mathcal{T}_\Delta$ in order to ensure the 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 and 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 the block header stores the hash root of a 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 such as 32 or 64 bytes, all transactions have almost the same size even if linked raw data have large sizes or require different formats, such as ENF signals or multimedia recordings.

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 for which its 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 sites closest to the address of a chunk not only serve information about the content but actually host data [35]. All sites in the Swarm network use the Kademlia DHT protocol [36], which synchronizes chunks in a P2P manner, to ensure data persistence and redundancy.

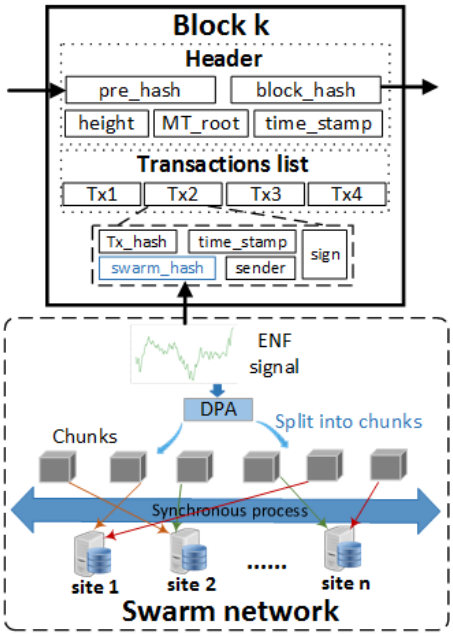

Figure 2. The illustration of block and off-chain data structure.

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

In this section, basic notations used in protocol design are clearly defined and explained. Then, an overview of PoENF consensus protocol is illustrated so that the reader can understand key components and workflow. Following that, we offer details on Byzantine resistant PoENF algorithms in block generation along with a voting-based chain finality. Finally, we also describe incentive mechanisms including rewards and punishment strategies given by mathematical analysis.

4.1. Basic Notation

Table 1 describes relevant notation used in PoENF model. To model sequential events in synchronous consensus rounds, a set of subsequential slots are used to define *Epoch*, which is represented as $sl_E = \{sl_1, sl_2, ..., sl_t\}$, 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 that 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 predefined collision-resistant hash function that outputs hash string $h \in \{0,1\}^{\lambda}$.

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

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

Definition 1. *Transaction is used to save data that are launched by a node u_i for recording on the distributed ledger, and its structure is represented as $tx = \{tx\_hash, pk_i, T_{stamp}, data, \sigma_i\}$, where the parameters are the following:*

- *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, such as swarm_hash or any byte strings;*
- *σ_i is a signature $RSA.sign_{sk_i}(tx\_hash, pk_i, T_{stamp}, data)$ signed by the sender's private key sk_i.*

Definition 2. *Block is a basic data unit that encapsulates valid transactions and is always appended on the chain head. A block generated at slot sl_t ($t \in 1, 2, 3, ..$) by validator v_j is represented by $B_i = (pre\_hash, height, mt\_root, tx\_list, sl_t, pk_j, \sigma_j)$, where the parameters are the following:*

- *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 the height of current block in blockchain (ledger);*
- *mt_root is a root hash of a Merkle tree of tx_list;*
- *tx_list is an orderly transactions list $[tx_1, tx_2, ..., tx_n]$;*
- *sl_t is a block created time stamp at the round sl_t;*
- *pk_j is public key of validator v_j;*
- *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 are organized as an ordered sequence of blocks starting 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 the following:*

- *$length$: 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;*
- *$head$: 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 proceed through those red nodes. The head of a blockchain is anchored on a recently confirmed block that 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 the credit weights of all nodes, K validators are randomly chosen to construct a 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.

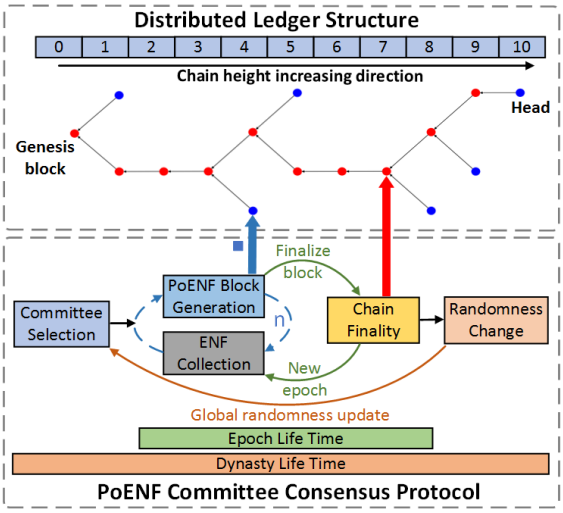

Figure 3. The PoENF consensus protocol overview.

For each epoch cycle, PoENF block generation and chain finality are core functions that ensure 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 the 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 by using a unique chain path.

At the end of current dynasty, PoENF committee members utilize the RandShare mechanism to cooperatively reach 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 the 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 the principles of chain-based Nakamoto protocols, the PoENF algorithm simulates a virtual mining method by pseudorandomly specifying a validator of committee as the slot leader to generate a block. To generate a block, a validator must present an ENF proof that 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 the 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 predefined 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 exist in the transactions pool;
(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 time slot. After the verification process, only valid transactions are cached as local transaction pools 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 the local transaction pool and removes outdated transactions that have 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 an 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 the *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 $k \leq K$.

In order to become a slot leader and propose a new block in the 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 the 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 the Krum aggregation rule to provide the (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 at most only f is sent by Byzantine nodes. For any $i \neq j$, let $i \to j$ denote the fact that E_j belongs to the $n - f - 2$ closest ENF proofs to E_i. Then, we define the ENF score for v_i.

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

Equation (1) calculates ENF scores $(s(1), ..., s(n))$ associated with validators v_1 to v_n, respectively, and applies the 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 the following.

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 it does, v_j wins the leader election and is qualified to propose a block; otherwise, the 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 the *generate_block*() function to propose a candidate block according to its collected ENF proofs in the 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 the current round finished. The closer s_i is to s^*, the higher probability that v_i can propose a new block.

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*

In block verification process, each validator calls the $verify\_block()$ 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 the 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.

4.3.3. Chain Extension Policies

In a PoENF block generation round, validators extend the 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 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;
   (b) For the candidate blocks having ENF proof transactions of the same size, the block generated by validator that 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 the current round. The chain head update process follows the rule i) $b = 1$.

Rule (i) covers a basic scenario to ensure that a new blocks is extended on the chain head. 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 the right to block proposal. Rule (iii) guarantees 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

Since fork issues are 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 are 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 to finalize 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 the already committed blocks on the main chain.

The chain finality protocol is mainly for identifying 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 *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 include performance improvement and security guarantees by EconLedger, this section briefly discusses incentive design while leaving detailed analysis for future investigation. 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 the current round. The incentive mechanism uses ENF score to evaluate a validator's contribution, and reward 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 reward rule is defined as follows:

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

where γ_i is the reward fee that v_i obtains from the total reward fees $\mathcal{R}$ during current block generation round. The smaller the ENF score of a validator, the higher the reward 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 honest contributors. As ENF fluctuations are randomly generated from power grids and vary at different times, 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 the credit stake c, the higher the 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 an 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, such as 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 fixed 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 the assumption that an adversary can only compromise no more than f nodes on the network, the slashing security deposit rule can increase financial cost if attackers use these compromised nodes to disturb consensus protocol, while reducing credit stake results in the lower probability that Byzantine nodes can be selected as committee members.

5. Experiment and Evaluation

In this section, a proof-of-concept prototype implementation and experimental configuration is described. Following that, we evaluate Econledger based on numerical results in terms of network latency, computation overhead, and communication throughput. Then, comparative experiments based on benchmark blockchain platforms are performed to show performance improvement. Finally, we analyze the performance and security properties provided by EconLedger.

5.1. Prototype Implementation and Experimental Setup

To verify the proposed EconLedger, a concept-proof prototype is implemented in Python, which consists of approximately 3100 lines of code. We adopted Flask [41], which is a light micro-framework for Python application, in order to implement networking and web service APIs for EconLedger node. All cryptographic functions are developed on the foundation of standard python lib: cryptography [42], such as 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.

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 devices can work as validators and perform the PoENF consensus protocol. Dell Optiplex 760 desktop functions as edge server, and five desktops are configured as sites in our private Swarm network. To initiate comparative evaluation between EconLedger and existing blockchain benchmarks, test cases are also conducted 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.

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), 3.2 GHz	Intel Core TM E8400 (2 cores), 3 GHz	Broadcom ARM Cortex A72 (ARMv8) , 1.5 GHz
Memory	8 GB DDR3	4 GB DDR3	4 GB SDRAM
Storage	350 G HHD	250 G HHD	64 GB (microSD)
OS	Ubuntu 16.04	Ubuntu 16.04	Raspbian (Jessie)

5.2. Performance Evaluation

In order 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 costs 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 requires $n \geq 3f + 1$, the minimum PoENF committee size is five members such that any $K \geq 5$ can meet both security requirements. Given 60 s 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 with respect to completing 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 tasks simultaneously and waited until the bundle of tasks is finished. The latency includes the round trip time (RTT) and service processing time on the remote host. 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 linearly 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 the 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 using Equation (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 value such as 60, and it has less effect on computation cost than K does. Thus, $\mathcal{T}_{bp}$ is almost linearly scaled relative to $\mathcal{O}(K^2)$. The total latency shows that EconLedger takes about 2 s to finish an epoch cycle of PoENF committee consensus $(\mathcal{T}_{ec} + \mathcal{T}_{bp} + \mathcal{T}_{cf})$ given the committee size $K = 20$.

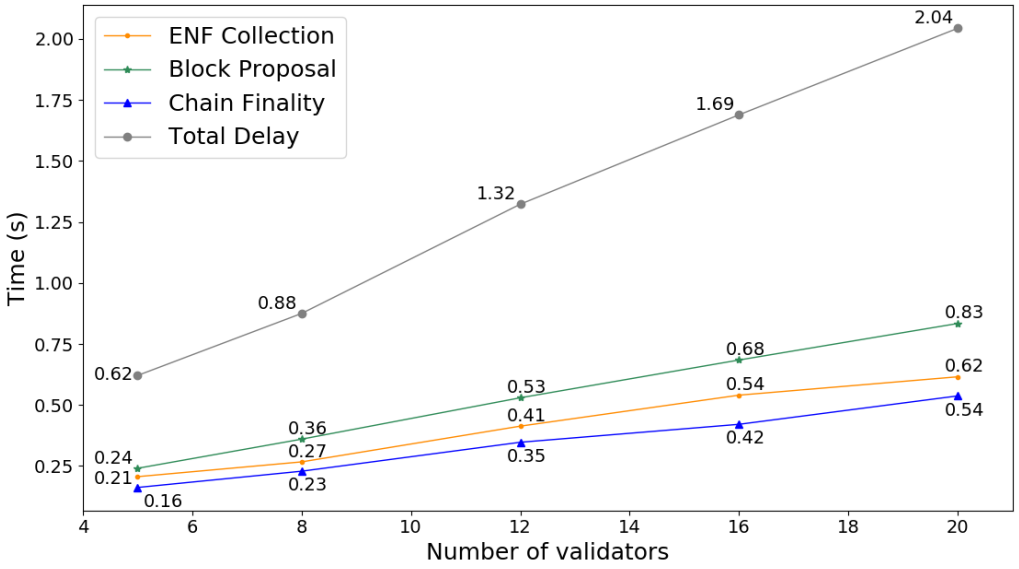

Figure 4. Latency for an epoch cycle of PoENF consensus with different committee size.

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, which simply checks the validity of a tx then buffers 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 verifying 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 K is scaled up. Given different computation capacity of benchmarks, RPi-4 needs 2.5× processing time than Desktop does.

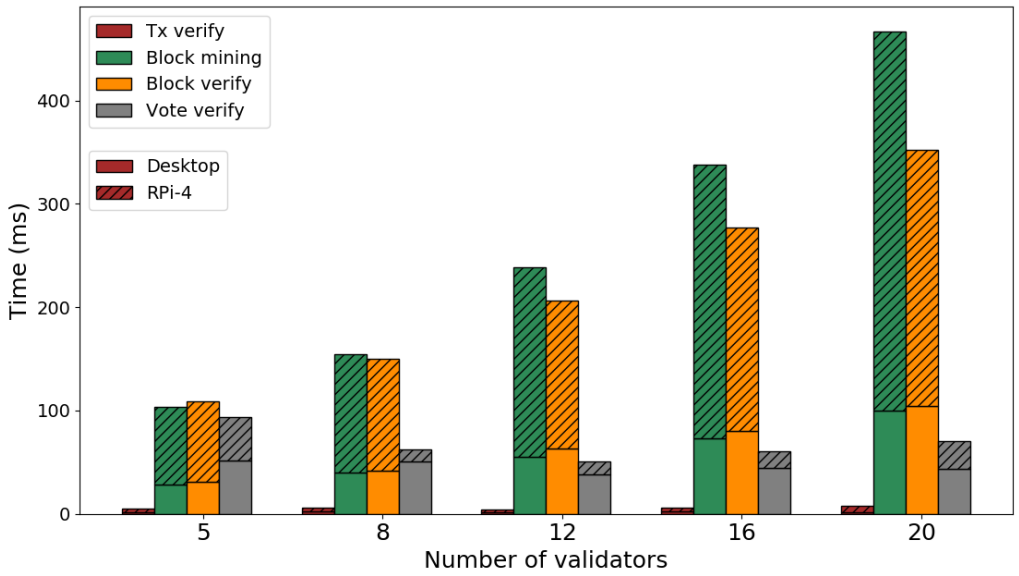

Figure 5. Computation overhead for stages of running PoENF consensus on host. Comparative evaluations on platform benchmark.

5.2.3. Data Throughput

In order to evaluate overhead of running EconLedger on communication channel, we considered volumes of message propagation and data throughput during key steps of the 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 the total communication complexity of $\mathcal{O}(K^2)$ in both ENF collection and chain finality, data transmission of ENF collection is $\mathcal{D}_{ec} = d_{tx} \times K^2$, and the data transmission of chain finality is $\mathcal{D}_{cf} = d_{vt} \times K^2$. Thus, communication overheads incurred by ENF collection and chain finality are linearly scaled to K^2.

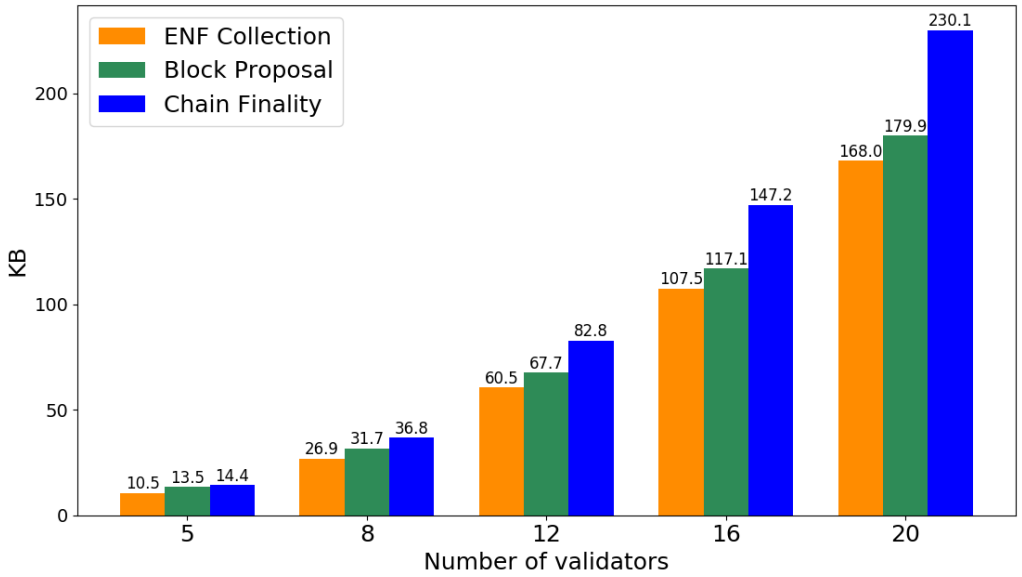

Figure 6. Communication overhead of running an epoch round of PoENF consensus. Comparative evaluations on different committee size.

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 obtain block size $d_B = d_{head} + d_{txs}K$. Therefore, the block size d_B is linearly scaled to K. Assuming an ideal case that only one valid block is proposed during each epoch cycle, data transmission of block proposal is $\mathcal{D}_{bp} = d_B \times K = d_{head}K + d_{txs}K^2$ such that communication overhead is almost scaled to K^2. On the other hand, for the worst case that every validator proposed a candidate block such that $\mathcal{D}_{bp} = d_B \times K^2$, huge communication cost scaling up K can be introduced.

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, the 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 a majority of LAN-based IoVT systems.

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

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 used smart contract to record transactions on blockchain. For the Tendermint network, we used the built-in kvstore as an ABCI (Application BlockChain Interface) app to save transactions.

We conducted 50 Monte Carlo test runs, where a node sends a 1KB tx per second (TPS) to the blockchain network and waits until tx has been confirmed on the 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 a red dot. The gray line shows the entire data range, and the black star is the median. Tendermint uses a BFT consensus protocol to achieve high efficiency; therefore, the mean of tx committed time is about 3 s given one voting round per second. Unlike Tendermint, Ethereum relies on probabilistic PoW consensus, which has variable block confirmation times. Thus, tx committed time in the Ethereum network varies with largest standard deviation. To guarantee synchronous epoch rounds for PoENF consensus, we set $\mathcal{T}_\Delta$ conservatively to 2 s based on the 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).

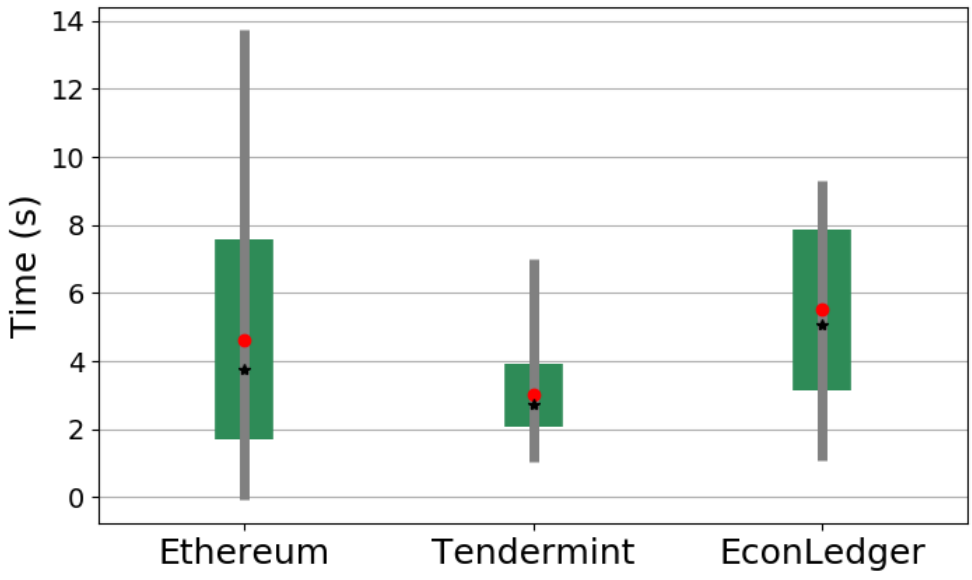

Figure 7. Time latency for committing transactions. Comparative evaluations on different blockchain networks.

Table 4 provides a comprehensive performance of committing transactions on different blockchain networks regarding several key performance matrices. Given the above tx committed time, which uses the mean in Figure 7, the tx rate tx/s is evaluated by calculating how many tx can be processed per second in the blockchain network. The 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 20 July 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 obtain the tx rate as $(571.4/4.6) \approx 124$ tx/s. Tendermint and EconLedger both use fixed 1MB block. Given 1 KB per transaction, a block in Tendermint can store a maximum of 1000 transactions; thus, the tx rate is about $(1000/2.9) \approx 344$ 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 at around $(2400/5.5) \approx 436$ tx/s.

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)	1200	72	80

In order to evaluate resource consumption by running blockchain benchmarks, we used the "top" command to monitor system performance of machines. We considered 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 1.2 GB 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 terms of running time. However, EconLedger has the higher *tx* committed time than Tendermint does, and it only needs 40% of the computation resource that Tendermint does.

5.4. Performance and Security Analysis

This section analyzes performance improvements given by the above experimental results and highlights the advantages of EconLedger compared with existing consensus protocols. Then, we evaluate security guarantees regarding committee selection and consensus algorithm. Finally, we list possible attacks and explain how EconLedge can prevent or mitigate these potential risks.

5.4.1. Performance Improvements

Given the above numerical results in terms of processing time and running time resource usages, our PoENF consensus is more computationally efficient than the PoW-based methods. Such a lightweight property of PoENF is promising for reducing energy consumption in mining processes and can lower demands on system capability for participants. Thus, resource-limited IoVT devices can directly work as validators (miners) rather than depending on support from an intermediate consensus layer by outsourcing mining tasks on fog networks or cloud servers. Compared with these hardware dependent solutions, such as REM based on Intel SGX and PoR requiring 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 levels of data transfer overhead on IoVT systems at the network of edge, which has limited bandwidth. In addition, communication complexity for each validator is linearly scaled to PoENF committee size, as shown in Figure 6. Thus, limited data transmission also means lower energy consumption on devices during communication handling tasks. Unlike non-scalable BFT-based solutions that rely on a pre-fixed set of validators, EconLedger aims to improve scalability by requiring a randomly elected consensus committee to delegate other nodes of the network. As a tradeoff, 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 a reference is a fixed length of hash value disregarding format or size of the source data, such light transactions can be used to verified complicated data in use over IoVT systems, such as multimedia recordings, contextual information, and trained models, etc. Moreover, each tx has fixed and small size such that a block can record more txs. As a result, the txs rate increased given that the block confirmation time is stable.

5.4.2. Committee Randomness Security

We assumed 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 other 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 owing to the unpredictability property of the VRF-based randomness string generation. In addition, given the assumption that an adversary can 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 the mining process by increasing investments on computation resource or owned coins. Moreover, the Krum rule adopted in ENF score calculation chooses $n - f - 2$ closet ENF proofs and precludes the $f - 1$ 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 accept valid blocks generated in the 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 the local chain header or reject all transactions by extending an 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 to 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 that has been finalized on the distributed ledger. In Econledger, a voting-based chain finality mechanism ensures the total order and persistence of data recorded on the distributed ledger. Thus, once a transaction is finalized in the checkpoint block, all other honest validators will work on the finalized main chain and disregard any double spending transactions from attackers.

2. Free-riding attacks: There is a possibility of free-riding attacks that some lazy nodes only gain benefits by using the security service without fulfilling their responsibilities in the EconLedger network, such as forwarding messages or submitting ENF proofs. The

punishment strategies can prevent against free-riding attacks by reducing credit stake of dishonest nodes or even isolating them from the entire 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 the 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 attacks by sending duplicated ENF proofs. As ENF fluctuations of power grid vary as time changes, the 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 attacks 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 in order to achieve safety and liveness. By using on-chain ledger and DDB enabled off-chain storage, the EconLedger network reduces storage overheads on validators and guarantees security and resilience of data sharing in a distributed IoVT network. The experimental results based on a prototype demonstrate that it achieves higher computation efficiency and tx throughput than benchmarks.

The experimental results on the prototype are encouraging, but there still are open issues to solve before developing a practical solution in real-world video surveillance systems. Using ENF signals for proof of work in consensus process is creative, however, whether ENF variation extracted from multimedia is reliable given attacks on ENF recordings such as synchronizing ENF and injecting into raw video/audio data or colluding among adversaries by sharing ENF data, is still an open question. Thus, our ongoing efforts include validating the proposed architecture in a real-world video streamscontext, simulating attack scenarios such as using AI enabled methods to generate fake ENF recordings, and ensuring overall efficiency and security.

In addition, validators in EconLedger system cannot directly obtain cryptocurrency rewards though PoENF consensus, but they can gain benefits from transaction fees. As a punishment strategy, slashing security deposits can increase financial cost if the adversary uses sybil nodes to disturb consensus protocol. However, there are open questions on the incentive mechanism. Our future work will use game theory to evaluate how incentive mechanisms can enhance system robustness and security.

Moreover, our EconLedger solution aims to provide a lightweight and security distributed ledger under a small-scale IoVT network, such as a campus. However, it still requires more investigation on how to apply EconLedger at a large-scale application scenario, such as smart cities or smart grids. Another future investigation for our team is designing scalable blockchain infrastructure that relies on a hierarchical framework in order to federate multiple privately distributed ledgers.

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.

Funding: This research was funded by National Science Foundation, grant number CNS-2039342 and United States Air Force Office of Scientific Research, grant number FA9550-21-1-0229.

Data Availability Statement: Not Applicable, the study does not report any data.

Acknowledgments: 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.

Conflicts of Interest: The authors declare no conflict of interest.

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. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 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. In Proceedings of the 2018 IEEE International Smart Cities Conference (ISC2), Kansas City, MO, USA, 16–19 September 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. In Proceedings of the 4th ACM/IEEE Symposium on Edge Computing, Washington, DC, USA, 7–9 November 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 Commun. Surv. Tutor.* **2018**, *21*, 1676–1717. [CrossRef]
5. Nikouei, S.Y.; Chen, Y.; Faughnan, T.R. Smart surveillance as an edge service for real-time human detection and tracking. In Proceedings of the 2018 IEEE/ACM Symposium on Edge Computing (SEC), Seattle, WA, USA, 25–27 October 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. [CrossRef]
7. Nakamoto, S. *Bitcoin: A Peer-to-Peer Electronic Cash System*. Technical Report, Manubot. 2019. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 21 September 2021).
8. Miller, A.; Juels, A.; Shi, E.; Parno, B.; Katz, J. Permacoin: Repurposing bitcoin work for data preservation. In Proceedings of the 2014 IEEE Symposium on Security and Privacy, Berkeley, CA, USA, 18–21 May 2014; pp. 475–490.
9. Castro, M.; Liskov, B. Practical Byzantine fault tolerance. *OSDI* **1999**, *99*, 173–186.
10. Hajj-Ahmad, A.; Garg, R.; Wu, M. ENF-based region-of-recording identification for media signals. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 1125–1136. [CrossRef]
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. [CrossRef] [PubMed]
12. Swarm. Available online: https://ethersphere.github.io/swarm-home/ (accessed on 2 January 2020)
13. Bollen, M.H.; Gu, I.Y. *Signal Processing of Power Quality Disturbances*; John Wiley & Sons: Hoboken, NJ, USA, 2006; Volume 30.
14. Grigoras, C. Applications of ENF analysis in forensic authentication of digital audio and video recordings. *J. Audio Eng. Soc.* **2009**, *57*, 643–661.
15. Hajj-Ahmad, A.; Garg, R.; Wu, M. Instantaneous frequency estimation and localization for ENF signals. In Proceedings of The 2012 Asia Pacific Signal and Information Processing Association Annual Summit and Conference, Lanzhou, China, 18–21 November 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 Trans. Secur. Saf.* **2021**, *2018*, 168648.
17. Buterin, V. *A Next-Generation Smart Contract and Decentralized Application Platform*. White Paper. 2014. Available online: https://blockchainlab.com/pdf/Ethereum_white_papera_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf (accessed on 21 September 2021).
18. Juels, A.; Kaliski, B.S., Jr. PORs: Proofs of retrievability for large files. In Proceedings of the 14th ACM Conference on Computer and Communications Security, New York, NY, USA, 29 October 2007; pp. 584–597.
19. Zhang, F.; Eyal, I.; Escriva, R.; Juels, A.; Van Renesse, R. REM: Resource-Efficient Mining for Blockchains. In Proceedings of the 26th USENIX Security Symposium (USENIX Security 17), Vancouver, BC, Canada, 16–18 August 2017; pp. 1427–1444.
20. Sawtooth Documentation. Available online: https://sawtooth.hyperledger.org/docs/core/releases/latest/introduction.html (accessed on 2 January 2020).
21. King, S.; Nadal, S. *Ppcoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake*; Self-Published Paper; 2012; Volume 19. Available online: https://people.cs.georgetown.edu/~clay/classes/fall2017/835/papers/peercoin-paper.pdf (accessed on 21 September 2021).
22. Lamport, L.; Shostak, R.; Pease, M. The Byzantine generals problem. *ACM Trans. Program. Lang. Syst. (TOPLAS)* **1982**, *4*, 382–401. [CrossRef]
23. Hyperledger Fabric. Available online: http://https://github.com/hyperledger/fabric (accessed on 2 January 2020).
24. Buterin, V.; Griffith, V. Casper the friendly finality gadget. *arXiv* **2017**, arXiv:1710.09437.
25. Bao, Z.; Shi, W.; He, D.; Chood, K.K.R. IoTChain: A three-tier blockchain-based IoT security architecture. *arXiv* **2018**, arXiv:1806.02008.
26. Tuli, S.; Mahmud, R.; Tuli, S.; Buyya, R. Fogbus: A blockchain-based lightweight framework for edge and fog computing. *J. Syst. Softw.* **2019**, *154*, 22–36. [CrossRef]
27. Sagirlar, G.; Carminati, B.; Ferrari, E.; Sheehan, J.D.; Ragnoli, E. Hybrid-iot: Hybrid blockchain architecture for internet of things-pow sub-blockchains. In Proceedings of the 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), Halifax, NS, Canada, 30 July–3 August 2018; pp. 1007–1016.
28. IOTA Foundation. IOTA Data Marketplace. Available online: https://data.iota.org (accessed on 2 January 2020).
29. Popov, S. The Tangle. 2018; p. 131. Available online: https://assets.ctfassets.net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3.pdf (accessed on 21 September 2021).
30. Xu, R.; Chen, Y.; Blasch, E.; Chen, G. Exploration of blockchain-enabled decentralized capability-based access control strategy for space situation awareness. *Opt. Eng.* **2019**, *58*, 041609. [CrossRef]
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. [CrossRef]
32. Nagothu, D.; Schwell, J.; Chen, Y.; Blasch, E.; Zhu, S. A study on smart online frame forging attacks against video surveillance system. In *Sensors and Systems for Space Applications XII*; International Society for Optics and Photonics: 2019; Volume 11017, p. 110170L. Available online: https://arxiv.org/pdf/1903.03473.pdf (accessed on 21 September 2021).

33. Xu, R.; Nagothu, D.; Chen, Y. Decentralized video input authentication as an edge service for smart cities. *IEEE Consum. Electron. Mag.* **2021**. [CrossRef]
34. Lamport, L. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM* **1978**, *21*, 558–565. [CrossRef]
35. Swarm Docs. Available online: https://swarm-guide.readthedocs.io/en/latest/introduction.html (accessed on 2 January 2020).
36. Maymounkov, P.; Mazieres, D. Kademlia: A peer-to-peer information system based on the xor metric. In *International Workshop on Peer-to-Peer Systems*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 53–65.
37. Gilad, Y.; Hemo, R.; Micali, S.; Vlachos, G.; Zeldovich, N. Algorand: Scaling byzantine agreements for cryptocurrencies. In Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, 28–31 October 2017; pp. 51–68.
38. Stadler, M. Publicly verifiable secret sharing. In *International Conference on the Theory and Applications of Cryptographic Techniques*; Springer: Berlin/Heidelberg, Germany, 1996; pp. 190–199.
39. Blanchard, P.; Guerraoui, R.; Stainer, J. Machine learning with adversaries: Byzantine tolerant gradient descent. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 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: Berlin/Heidelberg, Germany, 2021; pp. 129–149.
41. Flask: A Pyhon Microframework. Available online: http://flask.pocoo.org/ (accessed on 2 January 2020).
42. Pyca/Cryptography Documentation. Available online: http://pyca/cryptography (accessed on 2 January 2020).
43. SQLite. Available online: https://www.sqlite.org/index.html (accessed on 2 January 2020).
44. Ethereum Homestead Documentation. Available online: http://www.ethdocs.org/en/latest/index.html (accessed on 2 January 2020).
45. Kwon, J. Tendermint: Consensus without mining. *Draft Fall* **2014**, *1*, 11.
46. What's the Maximum Ethereum Block Size? Available online: https://ethgasstation.info/blog/ethereum-block-size/ (accessed on 2 January 2020).