Privacy-Preserving Framework to Facilitate Shared Data Access for Wearable Devices

Dane Troyer, Justin Henry, Hoda Maleki, Gokila Dorai, Bethany Sumner, Gagan Agrawal, Jon Ingram
Department of Computer & Cyber Sciences
Augusta University, Augusta, GA, United States
dtroyer, juhenry, hmaleki, gdorai, bsumner, gagrawal, joingram@augusta.edu

2021 IEEE International Conference on Big Data (Big Data) | 978-1-6654-3902-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/BigData52589.2021.9671690

***Abstract*—Wearable devices are emerging as effective modalities for the collection of individuals' data. While this data can be leveraged for use in several areas ranging from health-care to crime investigation, storing and securely accessing such information while preserving privacy and detecting any tampering attempts are significant challenges. This paper describes a decentralized system that ensures an individual's privacy, maintains an immutable log of any data access, and provides decentralized access control management. Our proposed framework uses a custom permissioned blockchain protocol to securely log data transactions from wearable devices in the blockchain ledger. We have implemented a proof-of-concept for our framework, and our preliminary evaluation is summarized to demonstrate our proposed framework's capabilities. We have also discussed various application scenarios of our privacy-preserving model using blockchain and proof-of-authority. Our research aims to detect data tampering attempts in data sharing scenarios using a thorough transaction log model.**

***Index Terms*—Security, Privacy, Blockchain, Health data sharing, Wearable Device**

I. Introduction

The introduction of the Internet of Things (IoT) implies that a litany of sensor input and raw data on individuals is available for processing and interpretation. Wearable IoT devices collect data on the individuals with the primary aim of helping individuals (and their healthcare providers) monitor their health and activity levels. As the time for direct interactions between an individual and their clinical service provider(s) is limited, the wearable data recorded over a period of time from the patient will aid the physician to accurately understand the activity levels and potential abnormalities.

Analytics on such data can have other benefits. For example, with health-related data collected through a wearable device, a less obvious, but no less important, alternate use might be a criminal investigation and producing a timeline of user activities [1]–[7]. Previous literature [8] states that complex analysis of wearable device data can lead to critical evidentiary data discovery. If a suspect of a crime happens to be wearing such a device, the heart rate monitor may indicate a level of physical activity that may (or may not) exonerate the individual. Not surprisingly, with such an unexpected use of this data modality, there is a heavy ongoing debate going on about the admissibility of wearable device data for use in court and trials [9]–[11].

In these big data applications that deal with high velocity and heterogeneity of data, a veracity related challenge is *data integrity*, which in this context means preventing unauthorized accesses or unauthorized modifications of data. In recent years, blockchain has been used as a means to provide trusted computing and authenticity [12]. This paper provides a proof-of-concept of a framework that uses permissioned blockchain to conduct (and establish proof of) secure transfer of data between entities. Permissioned blockchain [13] is a type of blockchain where all the nodes will be identified by the system before joining the system so that certain actions can be performed only by identifiable participants. A blockchain-based data access system will make big data more valuable because it ensures data quality, accessibility, and security.

Using a blockchain to store and determine authenticity for a database renders modifying that database with false transactions computationally infeasible. The way that a blockchain accomplishes this is through consensus, which is the mechanism through which transactions on a blockchain are authorized. Our framework uses the Proof of Authority consensus model to validate transactions on our database. We refer to our framework as PPAM or *Permissioned Proof of Authority Model*.

At a very high level, our framework takes the user's wearable device data, then preprocesses it using our parser/preprocessor, and stores the data in the database. It also stores access permissions that the user provides, data digest, and some auxiliary information for tracking purposes on the blockchain network. Whenever another user, who is not the data owner, tries to access the user's data, the framework will first check the permission list and the requestor's identity and then the permission will be granted accordingly. At the same time, it records the access request in the blockchain.

In all, the PPAM framework is a general model that can be used for sharing wearable data, verifying the integrity of shared data, and manage access control list. In the two examples discussed above, this framework can be used, as examples, in the following scenarios:

- A patient may want to share his/her health data tracked in their wearable device with a clinical service provider and then that data can also be shared with various other health service providers.
- A user may want to share the data with an insurance company in a scenario where an insurance company requests the user's health data to analyze their activity record in order to process an injury-related claim.

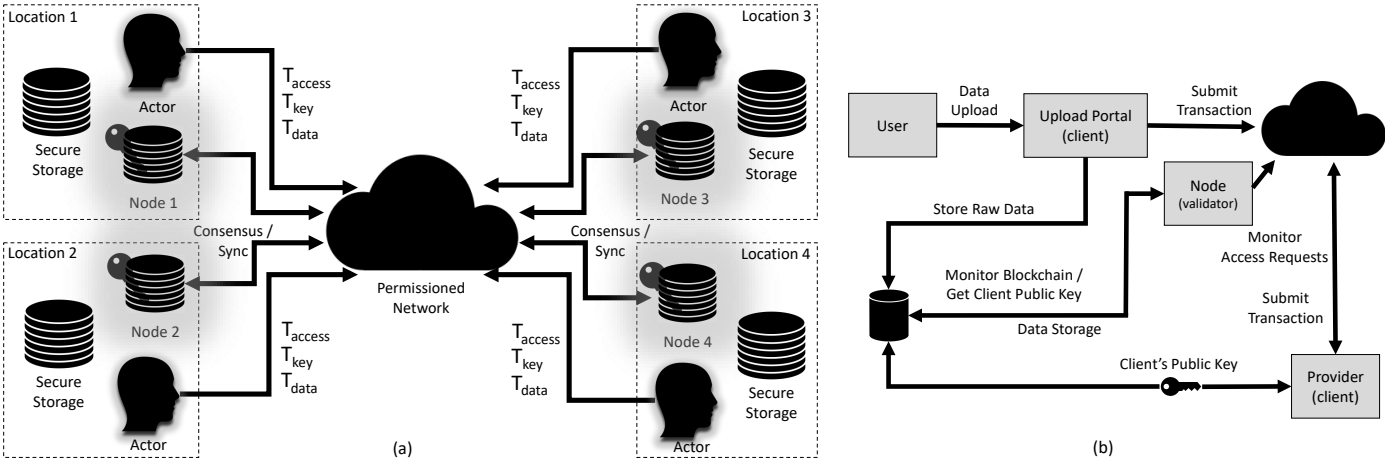

Fig. 1. Large scale PPAM framework example. Note that clients, nodes, and secure storage in each separate location assume peer-to-peer communication amongst themselves.

- A user may want to share the data with an insurance company in a scenario where an insurance company may want access to the user's health and medical information if the user is a victim of a hit-and-run or a car accident case.
- A scenario where a bystander or a victim wants to share some evidentiary data (text/picture/video/health-activity data) in a secure/private manner for digital investigation purposes.

Such a system can also help agencies meet requirements of regulatory frameworks. For example, in order to safeguard EHR data and protect user privacy, there are a few popular regulations such as, Health Insurance Portability and Accountability Act (HIPAA) [14] and the General Data Protection Regulation (GDPR) [15]. Some of the suggested measures [16] as laid out by HIPAA are to use access control, encrypt stored information and to add a feature of audit trail. Our PPAM framework can assure that the wearable health data shared by a user with their practitioners will be protected from intentional or unintentional tampering using immutable logs.

The rest of this paper is organized as follows: In Section-II, we present our PPAM framework, its overview, our algorithms and various components of PPAM. In Section-III, we discuss the framework protocol, data sharing and integrity measures. We have presented our preliminary evaluation in Section-IV. In Section-V we provide a summary of previous literature and related works. In Section-V-A we have discussed the limitations of existing works that motivated the current research and provided a comparative analysis of PPAM with other recent related works. Finally we conclude our work mentioning about the future research directions and conclusion in Sections-VI to VII.

II. PPAM Framework

Our description of the framework consists of three entities (as also shown in Figure-1(a)), which are *actor*, *node*, and *database*. *Actor* is an entity who is interested in storing, accessing, or manipulating data and interacts with our application. An actor can be either a *user*, who is interested in storing or retrieving their own data in the database, or a *provider*, who processes the data for operational reasons. A *node* is one of the entities that runs the blockchain protocol. *Database* stores the user (and potentially provider's) data, and in our system, is a private key-value data store. It interacts with the blockchain when the data is stored or read to correlate data being used by the blockchain for the purpose of consensus. For example, when a user adds new information into the database, the database communicates with the blockchain to ensure that the value logged into the blockchain matches the value stored in the database.

Figure-1(a) represents a distributed model showing various location entities. And, a granular representation of any individual physical location and its low-level details are illustrated in Figure-1(b). This also illustrates the difference between the user, provider, client, and validator as well as how the node receives transaction information.

Blockchain participants, nodes, are broken into two categories: *client* and *validator* nodes. *Client* nodes only exist to generate transactions to be added to the blockchain, and are not considered to be trusted entities on the network. Clients can be many types of participants, both human and automated, and include service providers and client upload portal backends. Conversely, *validator* nodes exist as trusted entities that receive transactions from clients, validate them, and generate blocks to be added to the blockchain. Additionally, validator nodes act as a form of distributed Certificate Authority (CA), collectively managing who may participate in the network by controlling the addition of public encryption keys to the blockchain. All validator nodes are considered to be trusted, with the network requiring the identity of the their operating organization to be verified prior to that node being allowed to participate in the framework as a validator node. The validator nodes are considered physically-secured, stand-alone servers in our framework.

The framework is designed as follows. The blockchain accepts three type of transactions: T_{access}, used for data access control management; T_{data}, used for data storage and retrieval; and T_{key}, used for key authorization. When a user signs up for the first time, a new pair of the key, (pk, sk), is generated. Since the user communicates with the blockchain network through an untrusted client node , a the public key pk, together with some auxiliary data that identifies the user, is sent to both the provider and the blockchain in T_{key}. Note that public keys of nodes are stored on the blockchain, as a means of maintaining key-to-identity links, allowing validator nodes to control keys to be added via a separate, manual consensus process.

Next, the user's data is preprocessed. The preprocessed data is encrypted with provider's public key and is transferred to the provider together with the timestamp and the hash of the processed data. At the same time the pointer to the preprocessed data, that is the SHA-256 hash of the data, is sent to the blockchain as part of T_{data}. Finally, the user defines the associated permissions given to the provider and sends it to the blockchain as part of T_{access}.

After the initial steps, the actor can query the data using T_{data} transaction. If the data requestor is the provider, then blockchain has to verify the provider's digital signature and check the granted access permissions. Note that the user can revoke or change access permissions using T_{access} at anytime. Thus, the blockchain will look for the latest access permission provided by the user for that provider.

III. Framework Protocol

Here we describe the underlying protocol that is used in PPAM. We utilize RSA asymmetric cryptography for two key reasons: encryption and signature. Therefore, for clarity, for encryption we use the 3-tuple (Gen_{enc}, Enc, Dec) where Gen_{enc} is the generator, Enc is the encryption, and Dec is the decryption algorithms. For signature we use the 3-tuple (G_{sig}, Sig, Ver) where Gen_{sig} is the generator, Sig is the signature, and Ver is the verification algorithms respectively. We also utilize a cryptographic hashing (referred to going forward simply as a hash) function $\mathcal{H}$ using SHA-256. Blockchain memory $\mathcal{M}$ represents the entirety of information stored on the blockchain as a key-value structured database.

PPAM is capable of logging all interactions that clients have with databases inside the framework. As previously mentioned, the blockchain in PPAM serves as both the CA and access log mechanism. Actors generate transactions on client nodes to interact with the database. As they do, the client node queries the blockchain to ensure said actor has appropriate permissions to execute the requested action. The blockchain stores these transactions in blocks. Whenever a validator node in the blockchain detects that there are transactions that have not yet been mined to the blockchain they mine them and then attempt to reach consensus with the rest of the validators. If successful, a new block is created and distributed to the ledgers on each local node. That block will contain header information for both identification and security purposes. In the body of the block, the transactions that were just mined can be found. Once stored on the blockchain, a block cannot be modified, which ensures immutable logging. PPAM providing these logging and security functions ensures that whatever data is being passed over the network will remain secure, and the logging of that data will be recorded in such a way as to be considered immutable. In this paper, we use data that has been taken from wearable devices as the data we log and secure in PPAM.

A. Wearable Data Preprocessing

Health Data Parser: Smartphone devices are equipped with advanced sensors to monitor health and collect activity data from a user. When a smartphone is paired with a smartphone-compatible wearable device, it helps to track the user's activity without the need to carry the smartphone devices with them at all times. The health app on the smartphone syncs the data collected using the wearable device, makes it available to the user with advanced tracking abilities, and shows a summary of health/activity data over time. This data can be exported [17] and shared with external entities [18]. This data can also be made available to third-party applications running on the smartphone device for further analysis, and sharing ([19], [20]). Once the data is available in the health app, the user's health data from the wearable device synced with the smartphone is exported as an XML file. Figure-2 shows a brief representation of the data preprocessing.

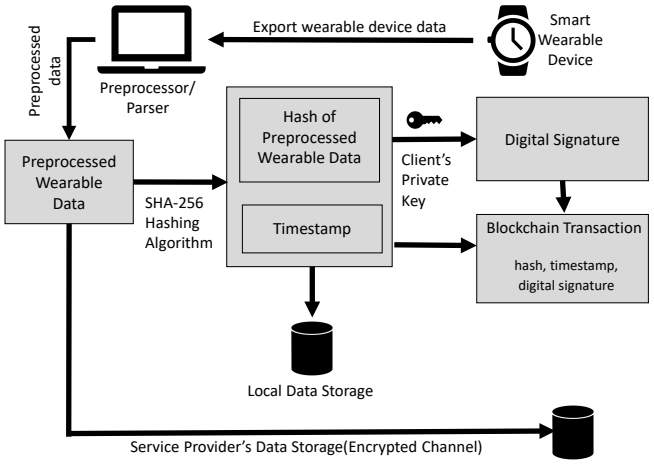

Fig. 2. An Overview of Wearable Data Parser, Cryptographic Hashing, Digital Signature and Blockchain Transaction Record Entry Process

Preprocessing Application: A python-based preprocessing application is used to extract the contents from the XML file's dictionaries. The parent element in the XML data is called as a *Record*. Our application begins to parse for a specific child element and grabs the data from the record. The record type, creation date, start date, end date, value, and unit are the parsing parameters. The preprocessing application is designed to parse the data based on the user's choice of a time interval. After parsing all the data, we convert it into Data Frame objects (a 2D labeled data structure with potentially

different types of columns). The output is a CSV file with preprocessed activity data. Our application specifically parses the *Sleep Analysis, Step Count, Stand Time, Stand Hour, and DistanceWalkingRunning* records. Note that sleep analysis is recorded by the apple watch using samples with overlapping times. In the next section, we will describe the PPAM framework and each of its components. The preprocessed data is primarily the input data to the PPAM framework.

As illustrated in Figure-2, the client's data from the wearable device is exported in the form of a XML file and then shared with the preprocessing application which will extract and parse only the subset of data related to step count, distance walked, activity related to standing and sleeping. Note that, for our proof-of-concept framework, we focus largely on the privacy-preserving model and hence processed only a subset of health/activity data instead of processing the major trove of health/activity data. At this stage, we use a CSV module to process the data within the preprocessed file. A python-based data dictionary reader used to match the information in each row of the CSV file with the respective fields. We use a python encoder to transcode the data into utf-8 format. This is a required step for the Blockchain hashing process. We then bind the API to the target function using route() [21] python decorator in Flask. Once the connection is made, the information from the client's (preprocessed wearable data) file will be visible.

In this implementation, the plain-text file containing the client's data is hashed using the SHA-256 cryptographic hashing algorithm, that provides an irreversible 256-bit unique hash. This hash, along with the file name, file path, and time accessed, are stored in a SQLite3 database. An RSA key object, which contains the private and public keys, is created using RSA's generate function. A signature is created to demonstrate the authenticity of the message, which is done using pkcs1_15, PyCryptodome's [22] signature scheme. The file is then re-hashed with SHA-256 algorithm on the recipient's side, which will be compared to the original hash made. Using a method to verify, we ensure the signature is valid and that the hashes from both sides match.

B. Blockchain Structure

At the most basic level, a *blockchain* is a way to record information. An apt descriptor that may assist in contemplating how a blockchain works would be to think of a blockchain as a ledger. What makes a blockchain special is that once information has been stored on the blockchain, it is computationally difficult to modify the block information. Thus we can say, once information exists on the ledger, it is permanent and immutable. While several blockchain frameworks exist currently, most require specific environments or programming languages to properly function. As such, we opted to build a permissioned framework from scratch, using the Python programming language.

Since this framework is intended to be used by a combination of organizations where collaborators and customers interact and exchange data, permissioned blockchain network is the best topology we recommend. In permissioned blockchain, only certain actions are allowed and all participants (i.e. nodes) should have an identity and permission to join the network. And we believe this model provides better data privacy.

In our model, all the submitted transaction must be digitally signed by the submitting actor's private key. This simple method ensures that the data originated with the actor and enables the receiver to detect alterations in the data. The validation of the transactions will be done by the validator nodes. The validators create blocks based on round robin algorithm. Once transactions are validated and added to a block and finalized by a validator node, this candidate block is proposed to the network of validator nodes for consensus. Once consensus is reached, all validator nodes add the block to their local blockchains and continue to propose subsequent candidate blocks based on additional transactions received during the consensus process. All blocks consist of header information used to guarantee data immutability and verification of the overall blockchain: previous block hash, block index, current hash, and the root of the transactions Merkel tree[1].

Proof of Authority (PoA) is a consensus algorithm that relies on a set of trusted nodes in a network to validate transactions on the blockchain. The most prominent PoA algorithms currently in use are Aura and Clique, both of which belong to Ethereum [24]. The benefits of PoA are many, but the ones most relevant to our implementation are scalability, efficiency, and ease of implementation. It is possible to scale our model in such a way that a blockchain using our model may have any number of v validators where $v > 2$. validators being the only nodes to compute consensus means that the other nodes in our implementation, clients, are free to submit transactions to the chain. In our consensus algorithm, only 51% of validators must verify a transaction.

Validator node consensus is achieved through several steps. Once a candidate block is mined by a validator node, it first passes the block directly to the next validator node in the network. Upon receipt, the consensus protocol is triggered on this validator node. The consensus protocol consists of two distinct phases occurring asynchronously across the network: local consensus and overall consensus. Overall consensus occurs once per node, while local consensus occurs on each validator node, once per additional validator node within the network.

Once consensus protocols are activated, the validator node first compares the received candidate block with its own most-recent block. If the block index is greater than the last local block, this candidate block is automatically accepted as "best" by this validator node. If the index matches (meaning there is a competing candidate block in the network), then a series of checks occur to determine best block. First, the number of transactions contained in the blocks are compared (longest block wins, Nakamoto Consensus). If the transaction quantity is identical between blocks, then block timestamps

[1]For more information on how to construct Merkel tree we refer the reader to [23].

are compared, with the earlier timestamped block winning. Once the best block is determined locally, the validator node sends this block to every other known validator node within the network. When each validator node receives this best block, its own identical consensus protocol is triggered, repeating the process. Through several cascading rounds, eventually through attrition a singular block remains across the network, which leads to overall consensus being reached once a majority of nodes have reached this conclusion. Any validator node may trigger overall consensus once recognized, but the block must be confirmed as valid by each participating validator node. The node must confirm the block locally via rehashing prior to adding the block to its own local blockchain. Adding a new block ceases the consensus process on each validator node. If a validator node replaces its own last block with a better block reached through consensus, any unaccounted transactions are stripped from the invalid block and reappended to an unverified transaction list locally for inclusion on the next block mined.

While complex on paper, this process of mining and consensus rounds was achieved in our test environment on an average of 2-3 seconds per block after the mining process started, proving that processing requirements for the validator node network remain low.

C. Blockchain Protocols

We have two core protocols and one auxiliary function that are executed on the blockchain. Function $AccessCheck(.,.)$ illustrated in Algorithm 1, checks where the requestor has permission to access the actor's data.

Algorithm 1 Access Control Check

procedure ACCESSCHECK($m, \mathcal{M}$)
  $a \leftarrow 0$
  $pk_p, pk_c, hash_t \leftarrow Decompose(m)$
  $check \leftarrow \mathcal{H}(pk_c||pk_p)$
  if $hash_t \in \mathcal{M}[check]$ **then**
    $a \leftarrow 1$
  end if
  return a
end procedure

Protocol 1 is executed by the validator node in the blockchain network for T_{access}. As illustrated in this protocol, the T_{access} transaction can be used to grant or revoke access to a specific provider for specific data. This transaction contains a timestamp t that guarantees the freshness of the signature. Whenever the user grants new access, the hash of the data will be added to the list of data that the provider can access.

Protocol 2 is executed when T_{data} is received. In this protocol, we used short notation DB for accessing the off-blockchain databases. If the requestor is the data owner, the requestor can append more data or read the current one stored in the database. However, if the requestor is not the data owner, then the $CheckAccess$ function will assist in making access decisions.

Protocol 1 Access List Protocol

procedure MANAGEACCESSLIST($T_{access}, \mathcal{M}$)
  $a \leftarrow 0$
  $(A, Sig_{sk_c}(A)) \leftarrow Decompose(T_{access})$ where
  $pk_c, ph_p, hash_t, t, Acc \leftarrow A$
  if $Ver(T_{access})$ and t is current **then**
    if $Acc = Grant$ **then**
      $\mathcal{M}(\mathcal{H}(pk_c||pk_p)) \leftarrow \mathcal{M}(\mathcal{H}(pk_c||pk_p)), hash_t$
    else
      Remove $hash_t$ from $\mathcal{M}(\mathcal{H}(pk_c||pk_p))$
    end if
    $a \leftarrow 1$
  end if
  return a
end procedure

Protocol 2 Data Storage / Retrieval Protocol

procedure MANAGEDATA($T_{data}, \mathcal{M}$)
  $(A, Sig_{sk}(A)) \leftarrow Decompose(T_{data})$ where
  $pk_c, ph_p, hash_t, t, rw \leftarrow A$
  if $Ver_{pk_p}(T_{data})$ and t is current **then**
    $m = (pk_p, pk_c, hash_t)$
    if $CheckAccess(m, \mathcal{M})$ **then**
      (DB) **return** $data \leftarrow hash_t$
    end if
  else if $Ver_{pk_c}(T_{data})$ and t is current **then**
    if $rw = read$ **then**
      (DB) **return** $data \leftarrow hash_t$
    end if
  end if
  return 0
end procedure

D. Accessing the Ledger

The blockchain ledger can be accessed in one of two ways: either provided as a copy by the organization, or directly accessed by users (provided the organization has provided a portal interface, API, or network access to do so). Since the blockchain itself contains no sensitive data to either users or the network, it can be freely shared amongst all requestors without concern. Our testing framework contained an open API endpoint specifically for getting a local copy of each validator node's chain, which can be easily, safely, and legally implemented in production. Additionally, a lightweight and open-source verification program to confirm the validity of the blockchain is not sensitive, complex, or proprietary, and can be freely distributed. A program to accomplish this was implemented as a part of our testing, and proved to be computationally lightweight and simple by first comparing that all validator nodes' blockchains were identical, then going through the chain in sequence to confirm each block's current hash matched an actual rehash of the block.

E. Adding Node to the Blockchain

Given that the permissioned blockchain network requires trust, proof of identity and non-repudiation are a requirement for certain nodes. As such, nodes requiring a higher level of trust, such as validator nodes, have a higher threshold to participate in the network than clients. Our framework requires a minimum of 3 validator nodes to function properly - those 3 nodes are assumed to be trusted by default, as they are founding nodes for the network genesis with the first three blocks of the blockchain populated with their public keys. Additional nodes may be added as more organizations wish to participate in the network, with this addition not only requiring an invitation to the permissioned validator node network for logical network access but additionally a manual consensus of a majority of existing validator nodes and human review of the candidate validator's identity documents. These documents can be agreed to ahead of time by participating organizations (i.e. an Incorporation Document, or another legal document proving the identity of an organization). The candidate validator would join the network using credentials transmitted through another secure means, then immediately generate its RSA public-private key pair. The candidate validator would then perform basic network discovery of other nodes present on the network. While the candidate validator would not have rights to participate yet on the blockchain, the blockchain is designed to be an open product. As such, any requestor may query a validator node and receive a copy of that node's local blockchain.

When a validator node is initially created, it is the responsibility of the utilizing organization to manually assign the candidate validator a key and a signature. The validator's T_{key} is a tuple of timestamp,its public key, and signature. When the candidate validator then attempts to join the network it will record the network addresses of all discovered validator nodes present, and query each for a copy of their local blockchains and public key. Then, the candidate validator would perform a comparison of each chain to ensure validity and that all chains are identical, immediately discarding duplicates. Next, the candidate validator will search the received blockchain for validator node key additions, and compare those keys in the blockchain to keys received by the validator nodes. Once those keys are deemed as valid (i.e. they match), the candidate validator will digitally sign its own identity document with its private key, and encrypt using the recipient validator node's public key; followed by a submission of the encrypted and signed document to each discovered validator node. Upon receipt, each validator node will require a manual, human review of the identity documents - if the document properly decrypts, the signature is decrypted by the accompanying public key, and the identity document positively confirms the identity of the invited candidate node, then the validator node will propose to the network that the candidate node's public key be added to the blockchain. If the majority of validator nodes reach this same consensus, the block is added and the candidate node gains full permissions to participate in the network as a validator node.

Client submission, conversely, is much simpler due to the fact that clients do not require the same level of trust as validator nodes. The positive identity of each client is managed by the organization to which they belong, so a client merely needs to submit its public key to its local validator node and construct the T_{key} transaction and submit it to the blockchain.A client's T_{key} is slightly different, with the addition of the public key for its local validator node. This is done to maintain accountability of client identities as well as to help discover potentially malicious or invalid transactions submitted by a rogue client.

F. Security and Privacy Analysis

As stated in [25], the immutability of a blockchain is, generally speaking, the immutability of its recorded data and the conditional immutability of its system rules and properties. We have already gone into detail about ensuring immutability of our recorded data. By virtue of our data being immutable and our data consisting of access logs then, so too, is our data provenance immutable. The entire implementation is one giant chain-of-custody. Also, privacy in our model is assured because we never store actual data on the blockchain, only logging information about said data. This leaves us with the conditional immutability of our system rules and properties.

As stated previously, PPAM is designed to be implemented on a permissioned network. Thus, it is the responsibility of whatever entity or entities who use PPAM to implement, maintain, and securely control access lists for database privileges. As long as the utilizing entity or entities maintain conditional immutability then, by extension, PPAM provides proper access control, logs, and data provenance; all of which are immutable.

An adversary would need to compromise more than half of the validators simultaneously to compromise our model. In order to compromise our model, an adversary would need to somehow compromise more than half of the validators simultaneously. PPAM being compromised in this way is highly unlikely due to distributed permissioned network.

Within our framework it is important to note that the user is ultimately in control over their own data. PPAM is resistant to malicious actors posing as a legitimate user because it is uses digital signatures for data integrity and sender authenticity. If an adversary were to gain access to the blockchain ledger, the user's data privacy is preserved because the only data stored inside the ledger are hashes of data or log information.

Note that users can use different pair of keys for different data. This guarantees that only a small fraction of the data may get compromised in the event of an adversary obtaining one of the user's private key.

We argue that keeping the blockchain itself as transparent as possible affords maximum opportunities for outside parties to audit the validity of the blockchain and increase trust in the immutability of the data contained, while not compromising security or privacy. In short, the more identical copies of the blockchain that exist across various systems, the harder it is

for any single entity to alter the veracity of the data in a covert way.

IV. Preliminary Evaluation

Current limitations of PPAM are dictated by the resources available to the entity utilizing our model in a production environment. As long as the only thing our model stores are hashes of coded information with the other block parameters the implementation can be as great in scope as the utilizing entity has resources for. In a test environment we have successfully generated chains containing over one thousand individual blocks, some containing hashes for multiple transactions, inside of a JSON file less than one megabyte in size.

This test environment consisted solely of a single VDI server (Type-1 Hypervisor) with 64GB RAM and 6 CPU cores running at 3.4GHz, spread across 6 Debian 10 virtual machines (VM) running Python 3.9 interpreters running as validator nodes. Two clients were also implemented in this network utilizing identical hardware specifications to simulate transaction submissions, with each client sending a randomized transaction at aperiodic intervals of 1-3 seconds. The local network was implemented using virtual switches on each VM connected to an additional VM acting as a router running pfSense. Within this environment, average block generation times (including consensus rounds) were approximately 2-3 seconds each after transactions were submitted. Each VM was only given 8000MB of RAM and a single virtual CPU for testing. An additional VM was used as a monitor to verify transactions on the blockchain in our testing environment.

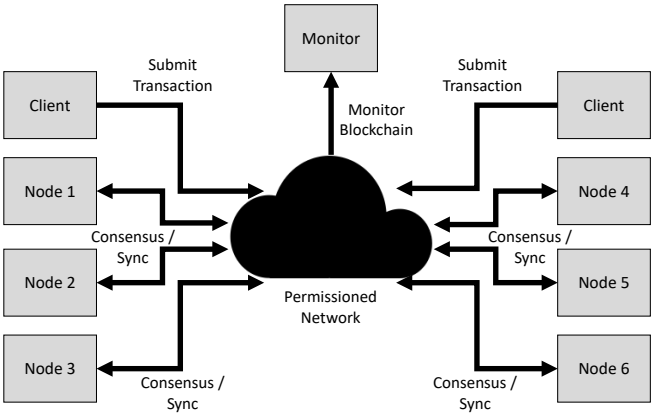

Fig. 3. The virtual environment used to test PPAM.

As illustrated in Figure-3 there is a difference between the test environment and the client node definition. In the test model, there is an absence of direct peer to peer communication between clients and validators. We were required to connect the client nodes to the entire network due to environmental constraints. Normally the clients would be connected to the validator nodes. We accounted for this and simulated the clients being connected to the validators during testing. Also, in an actual implementation, the monitor would not exist.

V. Related Work

Shi et al. [26] has produced a review of blockchain methods, and research opportunities for utilizing EHR (Electronic Health Record) systems. Along with the blockchain methods, their review also discusses the challenges and opportunities/benefits of blockchain usage. Properties such as autonomy, decentralization, security, pseudonymity, immutability, autonomy, incentive mechanism, and auditability are discussed as benefits a blockchain can provide for a EHR system. They also talk about blockchain-based storage and approaches related to proper data storage and identity management. Chenthara et.al [27] proposes a blockchain framework based on hyper-ledger fabric. According to the paper, their blockchain framework called "Healthchain" is used for efficient data sharing, health records management, and systematic access control. The Healthchain is considered patient-centric due to the patient's ability to completely control the healthcare record by delegating access and identity permissions. Our work explained in this paper uses a custom model to implement blockchain instead of hyper-ledger and a user will have the ability to provide access control for accessing their data (health data).

Li et al. [28] proposes a framework for forensic investigation known as IoTFC (IoT Forensic Chain). The overall goal of the research is to acquire potential digital evidence and mitigate investigation costs in the IoT environment. According to the paper, the IoTFC framework can achieve a comprehensive view of evidentiary items, provide continuous integrity, immutability and audit the tamper-proof environment with full provenance and traceability. Their method uses smart contracts and benefits from the following aspects: Autonomy (the ability to define the conditions to find related evidence items automatically); Trust (the evidence item can be encrypted on a shared ledger); Speed (the smart contracts can significantly reduce the examining time than manually process); Saving (smart contracts can save the cost without paying for middlemen, such as notary, witness) and Accuracy (the automated the smart contract runs in a faster, accurate, and cheaper way). While the IoTFC framework is applicable in a evidence extraction scenario for IoT devices, our PPAM framework work addresses a generic model applicable to various scenarios using blockchain protocol with proof-of-authority implementation.

Dorai et al.'s [8] research presents a forensic analysis framework for wearable devices called DEFA. DEFA is a proof-of-concept framework that incorporates visual and analysis tools for crime mapping and crime scene analysis queries. The objective of DEFA is to supply a solution to the hurdles digital forensic investigators face in both extracting and analyzing the sheer volume of extracted data. The DFEA framework desires to solve questions forensic investigations may have about how the extracted data should be stored and viewed, what analysis tools can help forensic investigators, and how the investigators can use various analysis techniques to get further insights into the culpability or not of suspects. The central portion of the paper discusses DFEA extracting data

TABLE I
A SUMMARY OF RELATED WORKS

Related Work	Summary
Chenthara et.al [27]	EHR and preserving privacy using Blockchain
Shi et al. [26]	Applications of Blockchain in EHR
Li et al. [28]	Framework for IoT digital forensics investigation using Blockchain
Dorai et al. [8]	Data extraction and forensic analysis for wearables
Lusetti et al. [29]	Custody of digital forensics in forensic medicine
Dimitrov et al. [30]	Healthcare data management using Blockchain
Wang et al. [31]	A new, pair-free scheme
Xiong et al. [32]	Large scale scheme for batch verification

from wearable devices/iPhones and its different analysis tools. Overall, their work uses statistical analysis for digital forensic analysis of wearable device data.

In Lusetti et al.'s work on a blockchain-based solution for the custody of digital files in forensic medicine [29], the framework implementation uses a distributed ledger model that keeps a record of transactions between two parties in a verifiable and permanent manner. Some of the problems the framework tries to address are storage handling, expectations of patients, decision-makers, and society, and the need for standardization of electronic evidence management. The blockchain model in this article is called the Custody Chain (CC). Their model uses two admin roles to manage the platform and grant data access: platform admins and case admins. The article states that platform admins are the information technology professionals in charge of operational functioning and user management. Case Admins are various registered users. The role depends on the jurisdiction, including healthcare professionals, prosecution officers, judges, lawyers, and other authorized professionals. While their research uses the model of admin roles to restrict/grant access to data in combination with a custody chain, we propose a consensus and proof-of-authority based model for storing, monitoring and sharing any kind of preprocessed data.

Dimitrov et al. [30] discuss various uses cases of blockchain. Some use cases are Electronic Data Management (EMR), Healthcare Data Management, Personal Health Record Data Management, Point-Of-Care Genomics, and EHR Data Management. The authors also discuss the various benefits of blockchain for EHR data management, which include enabling decentralized management, immutable audit trails, data provenance, robustness, and availability of data in order to increase the security and privacy of data. Similar use cases apply to our work, and the PPAM framework can achieve significant benefits such as decentralized management and immutable logs.

Wang et al. [31] analyzes existing certificate-less signature (CLS) protocols and proposes a new, pair free scheme that rectifies some of the more glaring security vulnerabilities in existing models centered around IoT devices. Vulnerabilities, such as MitM and DDoS attacks, are analyzed and proven to be less effective against their model. They also probe elliptic curve operations for cryptography. This work shares parallels with our PPAM framework in the form of issues we are trying to solve, most notably, robustness versus common exploitation methods.

Xiong et al. [32] reference Bitcoin's use of Elliptical Curve Digital Signature Algorithm (ECDSA) and claim that despite lowered computational cost, large scale batch sizes still impose too much cost to be practical. They address this issue with a large scale scheme for batch verification that outperforms current models. This research has potential use with PPAM as our framework has the ability to incorporate such a verification system with a low level of difficulty. This would produce a desirable outcome for our model, improving performance in large scale implementations, at a low cost.

A. Contributions of PPAM in Comparison to Other Recent Related Works

Our framework provides various advantages when compared to other current solutions. We identify four security risks that apply to existing models and compare them to our model as a tabular representation in Figure 4. We define these risks as Fault Tolerance, Man in the Middle (MitM), Privileged Attacker, and Malicious User. To expand upon these risks, we describe Fault Tolerance as the ability for the model to lose one or more nodes and continue to function. We describe MitM as an attacker intercepting network traffic and successfully gaining access to sensitive information. We define Privileged Attacker as an attacker from outside the system who gains access to legitimate credentials. We define Malicious User to be an authorized user of a system who attempts intentional or unintentional abuse of the system. PPAM provides several advantages over other contemporary solutions, especially in the instance of a malicious user. All of the compared recent works adequately handle Fault Tolerance and MitM attacks. The results of Wang et al. [31] and [28] lack a mechanism to identify whether or not a transaction that a node on their network has received originated from a valid node on their network. This makes them potentially vulnerable to an adversary who has enumerated their model and has the capability to send what appears to be a valid transaction. PPAM provides such a mechanism by way of being a permissioned blockchain. Jia et al. [33] provides an eloquent solution for dealing with privileged attackers by way of their improved certificate-less public key cryptography (CL-PKC) scheme. All of the cited works we compare to PPAM do not have a mechanism to identify exploitative actions by an authorized user. Whether those actions are intentional or not, PPAM can identify such

	Fault Tolerance	MitM	Privileged Attacker	Malicious User
PPAM (our solution)	✓	✓	✓	✓
Wang et al.	✓	✓	X	X
Jia et al.	✓	✓	✓	X
Li, Qin, and Min	✓	✓	X	X

Fig. 4. A comparison of our research to other work ([28], [31], [33].)

actions. As previously stated, PPAM acts as both the CA as well as an access log. In short, PPAM not only knows what actions which user takes, it also has the logs of user's actions, and those logs are immutable.

VI. Future Work

Future work on PPAM will entail testing in domains outside of health data as well as testing more robust implementations with access to greater computing resources. PPAM is suitable for any application where storing hashed data is the goal. We also theorize that PPAM would be suitable for applications where the entirety of the encrypted data could be stored on the blockchain. That is to say that PPAM could function as both the logger and the database. Further research into memory usage of this model is required before pursuing such an implementation. Finally, trust could be implemented as a variable on Validator nodes as a way to further decentralize this model [34].

VII. Conclusion

To conclude, the PPAM framework provides an organized model to store, share, and verify the integrity of wearable data collected from a user. PPAM is an easily adaptable privacy-preserving and data-sharing model for various applications and entities, including healthcare service providers, digital investigators, and other practitioners who want to analyze wearable device data. PPAM helps us meet the security and confidentiality requirements of health data sharing while preserving users' privacy.

Acknowledgment

We would like to formally acknowledge the contributions of a few key individuals who contributed to the completion of this project. We thank Hunter Thomas, a former research assistant, for his efforts in laying the groundwork for our PPAM framework model. We would also like to thank Prof. Konstantin Busch, Dept. of Computer & Cyber Sciences at Augusta University, who provided insights into various consensus algorithms.

References

[1] V. Waltz. (2015) Police: Florida woman fabricated rape report in lancaster county. FOX43.

[2] N. Y. Post. (2018) Victim's apple watch data used as evidence in murder trial. https://nypost.com/2018/04/02/authorities-used-apple-watch-data-to-identify-a-murder-suspect/.

[3] E. Moriarty. (2018) 21st century technology used to help solve wisconsin mom's murder. https://www.cbsnews.com/news/the-fitbit-alibi-21st-century-technology-used-to-help-solve-wisconsin-moms-murder/.

[4] S. Cole. (2018) Apple health data is being used as evidence in a rape and murder investigation. https://motherboard.vice.com/en_us/article/43q7qq/apple-health-data-is-being-used-as-evidence-in-a-rape-and-murder-investigation-germany.

[5] P. Olson. (2016) Fitbit data now being used in the court-room. https://www.forbes.com/sites/parmyolson/2014/11/16/fitbit-data-court-room-personal-injury-claim/#26c1d88b7379.

[6] A. Watts. (2017) Cops use murdered woman's fitbit to charge her husband. https://www.cnn.com/2017/04/25/us/fitbit-womans-death-investigation-trnd/index.html.

[7] C. Hauser. (2018) Police use fitbit data to charge 90-year-old man in stepdaughter's killing. https://www.nytimes.com/2018/10/03/us/fitbit-murder-arrest.html.

[8] G. Dorai, S. Houshmand, and S. Aggarwal, "Data extraction and forensic analysis for smartphone paired wearables and iot devices," in *HICSS*, 2020.

[9] N. Chauriye, "Wearable devices as admissible evidence: Technology is killing our opportunity to lie," *Catholic University Journal of Law and Technology*, vol. 24, no. 2, p. 9, 2016.

[10] S. Intille and A. M. Intille, "New challenges for privacy law: Wearable computers that create electronic digital diaries," *September*, vol. 15, p. 2003, 2003.

[11] T. R. Paranjpe. (2017) Social media, smart devices, and their use for trial strategies. http://www.texasbarcle.com/cle/OLViewArticle.asp?a=190461&t=PDF&e=15479&p=1.

[12] J. Evans. (2014) Bitcoin 2.0: Sidechains and ethereum and zerocash, oh my! https://techcrunch.com/2014/10/25/bitcoin-2-0-sidechains-and-zerocash-and-ethereum-oh-my/.

[13] H. Sukhwani, J. M. Martínez, X. Chang, K. S. Trivedi, and A. Rindos, "Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric)," in *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2017, pp. 253–255.

[14] HHS. (2021) The hipaa privacy rule. https://www.hhs.gov/hipaa/for-professionals/privacy/index.html.

[15] GDPR. (2020) What is gdpr. https://gdpr.eu/what-is-gdpr/.

[16] O. for Civil Rights. (2021) Understanding privacy and security of electronic records. https://www.hhs.gov/sites/default/files/ocr/privacy/hipaa/understanding/consumers/privacy-security-electronic-records.pdf.

[17] J. Padhiyar. (2017) How to export import health data in ios 10 in iphone. https://www.igeeksblog.com/how-to-export-import-health-data-in-ios-10/.

[18] O. Afonin. (2018) Extracting apple health data from icloud. https://blog.elcomsoft.com/2018/11/extracting-apple-health-data-from-icloud/.

[19] L. Peterson. (2016) How to export health app data on iphone and ipad. https://theappfactor.com/how-to-export-health-app-data-iphone-ipad/.

[20] F. Wahab. (2017) How to make sense of data exported from the ios health app. https://www.addictivetips.com/windows-tips/how-to-make-sense-of-data-exported-from-the-ios-health-app/.

[21] Pallets. (2021) View decorators. https://flask.palletsprojects.com/en/2.0.x/patterns/viewdecorators/.

[22] PyCryptodome. (2021) Pycryptodome documentation. https://pycryptodome.readthedocs.io/en/latest/.

[23] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[24] P. Ekparinya, V. Gramoli, and G. Jourjon, "The attack of the clones against proof-of-authority," *arXiv preprint arXiv:1902.10244*, 2019.

[25] F. Hofmann, S. Wurster, E. Ron, and M. Böhmecke-Schwafert, "The immutability concept of blockchains and benefits of early standardization," in *2017 ITU Kaleidoscope: Challenges for a Data-Driven Society (ITU K)*. IEEE, 2017, pp. 1–8.

[26] S. Shi, D. He, L. Li, N. Kumar, M. K. Khan, and K.-K. R. Choo, "Applications of blockchain in ensuring the security and privacy of electronic health record systems: A survey," *Computers & Security*, p. 101966, 2020.
[27] S. Chenthara, K. Ahmed, H. Wang, F. Whittaker, and Z. Chen, "Healthchain: A novel framework on privacy preservation of electronic health records using blockchain technology," *Plos one*, vol. 15, no. 12, p. e0243043, 2020.
[28] S. Li, T. Qin, and G. Min, "Blockchain-based digital forensics investigation framework in the internet of things and social systems," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 6, pp. 1433–1441, 2019.
[29] M. Lusetti, L. Salsi, and A. Dallatana, "A blockchain based solution for the custody of digital files in forensic medicine," *Forensic Science International: Digital Investigation*, vol. 35, p. 301017, 2020.
[30] D. V. Dimitrov, "Blockchain applications for healthcare data management," *Healthcare informatics research*, vol. 25, no. 1, pp. 51–56, 2019.
[31] W. Wang, H. Xu, M. Alazab, T. R. Gadekallu, Z. Han, and C. Su, "Blockchain-based reliable and efficient certificateless signature for iiot devices," *IEEE Transactions on Industrial Informatics*, 2021.
[32] H. Xiong, C. Jin, M. Alazab, K.-H. Yeh, H. Wang, T. R. R. Gadekallu, W. Wang, and C. Su, "On the design of blockchain-based ecdsa with fault-tolerant batch verication protocol for blockchain-enabled iomt," *IEEE Journal of Biomedical and Health Informatics*, 2021.
[33] X. Jia, D. He, Q. Liu, and K.-K. R. Choo, "An efficient provably-secure certificateless signature scheme for internet-of-things deployment," *Ad Hoc Networks*, vol. 71, pp. 78–87, 2018.
[34] G. Zyskind, O. Nathan *et al.*, "Decentralizing privacy: Using blockchain to protect personal data," in *2015 IEEE Security and Privacy Workshops*. IEEE, 2015, pp. 180–184.