Optimal Witnessing of Healthcare IoT Data Using Blockchain Logging Contract

Mohammad Hossein Chinaei, Hassan Habibi Gharakheili, and Vijay Sivaraman

***Abstract*—Verification of data generated by wearable sensors is increasingly becoming of concern to health service providers and insurance companies. These devices are typically vulnerable to a wide range of cybersecurity attacks, attempting to manipulate sensing data. Most of these disastrous attacks would remain undetected since neither healthcare servers nor IoT sensors are aware of the existence of attackers in the middle of communication. Thus, there is a need for a verification framework that various authorities can request a verification service for the local network data of a target IoT device. In this paper, we leverage blockchain as a distributed platform to realize an on-demand verification scheme. This allows authorities to automatically transact with connected devices for witnessing services. A public request is made for witness statements on the data of a target IoT that is transmitted on its local network, and subsequently, devices (in close vicinity of the target IoT) offer witnessing service.**

Our contributions are threefold: (1) We develop a system architecture based on blockchain and smart contract that enables authorities to dynamically avail a verification service for data of a subject device from a distributed set of witnesses which are willing to provide (in a privacy-preserving manner) their local wireless measurement in exchange of monetary return; (2) We then develop a method to optimally select witnesses in such a way that the verification error is minimized subject to monetary cost constraints; (3) Lastly, we evaluate the efficacy of our scheme using real Wi-Fi session traces collected from a five-storeyed building with more than thirty access points, representative of a hospital. According to the current pricing schedule of the Ethereum public blockchain, our scheme enables healthcare authorities to verify data transmitted from a typical wearable device with the verification error of the order 0.01% at cost of less than two dollars for one-hour witnessing service.

***Index Terms*—IoT, data witnessing, blockchain, optimization**

I. Introduction

MODERN healthcare systems are increasingly coming online. Wearable devices have profoundly improved patients experience of interacting with remote healthcare providers. They facilitate remote healthcare monitoring given their capability in automatic measurement of medical signs and periodic transmission of data over the Internet. Received medical data will be stored on a cloud server where different authorities can access data and take appropriate actions. Healthcare systems benefit from wearable devices to provide a higher quality of care to citizens, improved decision making, accurate real-time diagnosis, and timely treatment at considerably lower prices [1], [2]. A study in Australia [3] estimated that nearly $20,000 per patient can be saved annually by adopting tele-medicine healthcare. Many insurance companies have also developed policies based on the customers medical information received from wearable IoTs [4].

Automated and online health services driven by new wearable technologies, while revolutionizing the traditional health systems, come at a price of frequent and sophisticated cyber threats [5], [6], [7]. A range of security and privacy threats for IoT devices (medical and others) have been identified and analyzed by prior research works [8], [9], [10]. In the context of healthcare IoT systems particularity, "integrity of data" received from wearable sensors is of paramount importance to medical decisions made by practitioners as well as enhanced management of insurance claims.

Wearable device users, themselves, sometimes become potential attackers when they try to forge data transmitted from their health sensor to claim pecuniary benefits [11]. An adversary, even located at far physical distance from a target wearable device, can maliciously manipulate data packets transmitted from the sensor to falsify its sensitive information. Attackers may attempt to manipulate the data transmitted from a health sensor such as an insulin pump, causing the patient to receive a lethal dose of medicine [12]. More worryingly, most of these attacks would remain undetected as neither the victim device nor the healthcare server is aware of an adversary in the middle of their communications. The largely scattered distribution of IoT-dependent patients coupled with poor security measures embedded in devices make wearables even more appealing victims to attackers who target health data authenticity and integrity. This poses massive risks to health industry which increasingly demand "trust" for a variety of data records they maintain, especially those generated by patients.

Any discrepancy in data, between what is transmitted by the sensor and what is received by the healthcare system, warrants further investigation to check whether data is forged or tampered with. Therefore, obtaining the local version of data (*i.e.,* transmitted by wireless sensors) is crucial for verification. Given the high density of connected devices in smart environments [13], seeking help from other devices in the close proximity of the sensor or "witnessing" seems to be a viable approach [14].

Researchers have shown [14], [15] that wireless neighboring nodes (IoT and/or non-IoT devices) that share a wireless broadcast domain can overhear each others' data transmission, and hence have the potential to serve as witness. In [14], wireless network gateway and all connected devices are expected to record fingerprint of every data packet (per individual

M. H. Chinaei, H. Habibi Gharakheili, and V. Sivaraman are with the School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, NSW 2052, Australia (e-mails: m.chinaei@unsw.edu.au, h.habibi@unsw.edu.au, vijay@unsw.edu.au).

device) transmitted on the network. These records (witness logs) are sent to a central server on the Internet, allowing a forensic expert to retrospectively compare logs of the gateway against those of other connected devices for verifying the integrity of data transmitted by the subject sensor on a given network. Work in [14] primarily aimed to provide a compact and secure scheme for logging the overheard data packets in wireless sensor networks. The authors improved their method in a follow-up work [15] which provides witnessing location proof and preserves the privacy of witnesses. While proposing a novel idea, their scheme suffers from a number of issues: (a) network gateway serves two roles, namely an essential supplier of witnessing log and also the central node of the network that forwards the log of other witness devices, hence becoming a single point of failure if compromised; (b) it is impractical for IoT devices (typically resource-/energy-constrained) to continuously log every packet transmitted by each of the other connected devices – even wireless network gateways need to change and/or upgrade; (c) witnesses lack incentives to statically participate in an opportunistic collection of records for other sensors.

In this paper, we develop a new witnessing scheme that allows authorities to publicly request for witnessing of data transmitted by a target sensor, and incentivized witnesses to collect and submit their local records in a privacy-preserving manner. In order to develop a practical scheme for data witnessing and verification, certain requirements are to be met:

Secure and private logging: Witnesses record data packets from the target device and securely log them into their statements transmitted to and stored in a tamper-proof database for an admissible auditing. More importantly, statements should protect the privacy of witnesses (not revealing the identity and location of witnesses).

Dynamic witnessing: Witnessing incurs computing costs, and hence should be performed judiciously and dynamically as opposed to statically. The Health Service Providers (HSP) queries over a distributed service platform for witness statements corresponding to a target device whenever needed. The query is seen by potential witnesses that have subscribed to this service platform. Those witnesses which are in close vicinity of the target sensor may choose to contribute to this process by giving statements at certain resolution – higher resolutions incur heavier computing costs.

Monetization: Automated telehealth and/or telemonitoring systems have been proven to provide better quality and hassle-free medical services at considerably lower costs [16], [17]. However, HSPs are expected to spend the bulk of their budget on smart connected devices, and cloud/web services [3], [18], [19]. A fraction of this investment can be allocated to incentivize potential witnesses, covering their power and computing costs. A variety of options may be used for incentives such as monetary return (micro-payments per successful transactions), special tokens, or even discounts for premium services (similar to the plans offered by insurance companies to encourage their customers to have healthier lifestyles [20], [21]). HSPs potentially have the ability to incentivize potential witnesses; however, they need a systematic method that allows them to make dynamic decisions in choosing the right witnesses which meet their requirements while aligning with their budget. The HSP would select certain witnesses (from a set of available ones) that yield the highest verification (lowest error) subject to a budget constraint, and this selection can vary dynamically depending on the availability of witnesses in the environment and their ability in generating statements of certain resolutions.

Most of the previous works in the area of witnessing [22], [23], [14], [15] focused on the first requirement above (*i.e.,* secure logging) in a variety of scenarios, while the other two essentials have been overlooked or poorly addressed. In this paper we use blockchain technology as a platform that connects individual witnesses to health authorities, enabling both parties to communicate, interact, and transact dynamically. Recently, blockchain in conjunction with automatic executable programs (aka "smart contracts" [24]) has enabled a medium for trading assets between different parties. Providing an immutable, shared, and real-time ledger along with enabling transactions among individual peers motivate us to develop our witnessing scheme on a blockchain platform.

This paper describes a novel witnessing solution that allows healthcare authorities to remotely and dynamically verify the data received from health IoT sensors. Note that our witnessing scheme does not change the way data is collected by wearable sensors, transmitted on the network, and stored on the health cloud. The health data may be collected and stored in a centralized or distributed fashion. The primary focus of this paper is on the verification of data, close to the sender, by way of collecting witness statements in a distributed manner. Our aim is to verify whether or not the received data has been tampered in transit from the sender sensor to the health cloud server. We develop a witnessing smart contract on top of Ethereum blockchain that enables both parties, namely the potential witnesses co-locating with a telehealth sensor and the HSP, to trade their assets/services using programmable interfaces. Empowered by an immutable, shared, and real-time ledger, our witnessing scheme allows potential witnesses to participate in an on-demand witnessing protocol without compromising their privacy. In comparison to alternative solutions like static witnessing, our scheme is more practical by the use of conventional blockchain technology and considerably cost-effective due to its dynamic nature (duration and quality of service are configurable by the HSP, meeting their business models/budget constraints).

Our **first** contribution is to develop a system architecture based on blockchain and smart contract functions which enable authorities to dynamically avail a verification service for data of a subject health sensor from a distributed set of witnesses which are willing to provide (in a privacy-preserving manner) their local wireless measurement in exchange of monetary return. Our **second** contribution develops a method to optimally select witnesses in such a way that the verification error is minimized subject to monetary cost constraints. **Finally**, we evaluate the efficacy of our scheme using real Wi-Fi session traces collected from a five-storeyed building with more than thirty access points, representative of a hospital. According to the current pricing schedule of the Ethereum public blockchain, our scheme enables healthcare authorities to verify data transmitted from a typical wearable device with

the verification error of the order 0.01% at cost of less than two dollars for one-hour witnessing service.

The rest of the paper is organized as follows: §II describes prior work on witnessing, data logging, and blockchain use-cases in data provenance, and §III describes our solution approach that systematically and dynamically makes use of smart contracts for distributed witnessing. In §IV we describe our optimization framework to select witnesses, while in §V we evaluate the efficacy of our scheme via simulation. The paper is concluded in §VI.

II. Related Work

This paper lies at the intersection of three strands of research works including intrusion detection systems (IDS), secure logging, and application of blockchain/smart-contract technologies. First, we emphasize that our objective is not to develop IDS method for IoTs, but alarms of a network IDS can be used to trigger our process of data verification. Second, we look at secure logging methods to maintain audit trail and discuss witnessing protocols that are close to our work. Third, we highlight the application of blockchain and smart contract technologies in data provenance, access control management, and how our scheme differs from existing works.

A. IDS for IoTs

Intrusion detection systems monitor network traffic and often look for signature of malicious activities in traditional IT networks. IDS appliances enable network auditors to detect compromised devices or cyber-attacks [25]. Intrusion detection in IoT infrastructures slightly differs from IT networks given their purpose-built with limited set of functionalities and identifiable network patterns [26], [27].

In [28], we developed a network-based IDS for IoT devices using Manufacturer Usage Description (MUD) profile and software-defined networking (SDN) paradigm to translate formal behaviors of off-the-shelf sensors to static and dynamic flow rules that can be enforced to the network at run-time – traffic that conforms to these rules can be allowed, while unexpected (anomalous) traffic is inspected for potential intrusions. Both signature-based and anomaly-based IDS solutions employ a centralized engine (often embedded into the network) to detect volumetric attacks (*e.g.,* reflection/amplification, flooding, ARP spoofing, and port scanning) or unintended traffic. Witnessing scheme in this paper can benefit from an IDS by receiving alarms to initiate the process of data verification. Also, our witnessing engine, independent of the local network infrastructure, is managed from the cloud by health authorities.

B. Secure logging

Data logging in traditional systems is designed to record any event that changes the state of the system and enables authorities to reconstruct chains of events. In other words, audit records are used as digital evidence. An auditing architecture traditionally consists of three primary roles: dedicated devices that capture audit records, collectors which store these records, and an auditor who retrieves the collected data and retrospectively investigates the records to detect any suspicious activity. Various secure logging methods have been proposed in both capturing and storing phases to fulfill the requirements of logging audit records as admissible evidence [22]. In what follows we briefly discuss some of important works in this space.

In [29], authors consider a secure logging architecture in which audit logs are stored on an untrusted machine during storage phase. They proposed a scheme based on hash chains and evolving shared cryptographic keys to limit attackers ability in reading and altering the audit logs. The shared key in each epoch is derived from the hash of previous data logs. Therefore, even an attacker manages to break one of the shared keys, they only can change audit logs for that specific epoch. Any audit log manipulation is detectable because the shared keys of next epochs are made using the original version of the audit log. The major drawback of this method is that verifiers need to posses the shared key of individual epochs to verify the authenticity of audit logs. This issue has been addressed by Logcrypt [30], where the author proposed using public key encryption which enables the audit log to be signed by one entity and verified by anyone else without revealing the secrets.

Authors in [31], while pointing out vulnerabilities of the previous works, propose a secure logging architecture based on Forward-Secure Sequential Aggregate (FssAgg) authentication techniques. Their proposed scheme takes a private key, a message to be signed, and the aggregated signature computed up to a certain point in order to compute a new aggregate signature. They showed that this scheme provides better security while incurring less computational and communication overhead. All of these schemes (mentioned above) demand either predefined powerful audit generators in the vicinity of a target IoT sensor or high computing power on the sensor itself to digitally sign its audit logs and transmit them to verification authorities. Therefore, none of them seem practical in the context of IoT sensors given their energy and compute limitation.

1) Witnessing: To the best of our knowledge, witnessing (seeking assistance from neighboring nodes) has been mostly used for location proof systems. In [32], authors propose APPLAUS a privacy-preserving location proof system that enables authorities to verify the location of a node based on the proofs collected from a set of witnesses in the environment. All parties, including the verifier and witnesses, need to register with a certification authority (CA) to obtain secrets for signing the proofs. A registered node, equipped with Bluetooth technology, can obtain a location proof from individual available witnesses in the environment and forward these proofs to a central proof server from which an authorized verifier would retrieve location proofs. To protect the identity of various parties (nodes and witnesses) from each other and from the location proof server, the CA provides each user with a set of private/public key pairs. Authors in [33] propose an improved scheme named STAMP based on a unique pair of public/private key and commitment techniques instead of periodically changed pseudonyms. They also remove the untrusted central proof server to better detect collusion.

Work in [34] proposes WORAL as a witness oriented

provenance framework for secure location proofs of mobile devices. In their scenario, each physical region has a designated location authority, and a set of mobile users, each can become a volunteer witness for the a user who needs to prove her location. Witnesses provide notarization of a statement between the user and location authorities of an environment. The devices have local memory to store provenance information which is fully controlled by its user – part of (or the entire) information can be provided to applications that need location proofs. These prior works fall under the category of secure location proofs. In our work, potential witnesses (if any in the environment) provide health authorities with a compact version of local network data transmitted by the target IoT sensor. We do not employ static and predefined location authorities which can be very difficult and expensive to scale. Importantly, existing methods do not consider practical monetary incentives for witness devices.

Authors of [14] propose a witnessing algorithm for secure logging and forensics of medical data. This work takes advantage of overhearing data packets transmitted on wireless networks. Witnesses (neighboring devices that share a local wireless network gateway) log a fingerprint (in form of Bloom filters) of all packets exchanged between the gateway and a target sensor. Bloom filters are periodically uploaded onto a central forensic server, and the gateway also sends a list of available witnesses to the forensic server. Their scheme, while proposing a novel application of witnessing, comes with a number of challenges and shortcomings related to privacy and implementation. First, their scheme discloses the location of the gateway, witnesses, and the target sensor to a third party, compromising the privacy of all entities contributing to the scheme. Second, this protocol presents a single point of failure since the gateway can be compromised, and hence sends a falsified/inaccurate list of witnesses to the central server. Third, their scheme would hardly scale. It needs different entities such as manufacturers of witnesses devices, the Internet service provider, and verification authorities to pre-coordinate with each other. Fourth, the witnesses are not incentivized to contribute to this scheme.

In [15], authors improve the previous work by introducing pseudonymous witnessing scheme. They use spatio-temporal characteristics of wireless links between the gateway and witnesses to generate proof of location and a third-party anonymizing server to address the privacy concerns. While being a practical idea to ensure witnesses are within the physical proximity of the gateway, the pseudonymous witnessing poorly addresses privacy concerns as it still needs gateway and witnesses to share their identities and locations with a third-party server. Also, their witnessing scheme is still static, and difficult to scale with no incentives for witnesses to participate.

C. Blockchain

The emerging blockchain technology and smart contracts on top of it provide a shared, distributed database governed by a consensus algorithm, not a single authority. Majority of network nodes decide upon accepting new events and chain them to the rest of historical events. The cryptographic structure of the data stored on the blockchain and the enormous amount of power needed to reach an agreement between different peers makes the blockchain tamper-proof against adversaries. Decentralized control and immutability of the blockchain make it a viable approach to develop trusted systems [35]. Smart contracts are pre-defined contractual terms written in the form of digital scripts which are executed if certain conditions are met. While traditional blockchains like bitcoin support only cryptocurrency transactions, more advanced blockchain like Ethereum support smart contracts, providing individual users with the opportunity to self-enforce their protocols for trading/exchanging various assets or services [24].

One of the use-cases of blockchain in the literature is for access control management. Work in [36] uses blockchain to provide an access control framework for IoT devices. Their framework uses the transactions on the shared ledger to grant, get, revoke, and delegate access for IoT applications. They focus on the rights of individual users as owner of the data and use the shared ledger of the blockchain as an immutable, tamper-proof database which enables the user of the sensor to give access to different parties.

Blockchain, as a tamper-proof database, has unique capabilities to address various issues related to security and privacy of the IoT ecosystem. In a recent work [10], authors identified security and privacy threats for IoT applications at multiple layers of the protocol stack, and surveyed blockchain-based solutions developed by prior works for scalable management of access, trust, secure storage, authentication, and access control. Our IoT data verification scheme introduces a new application of blockchain in addressing concerns pertinent to integrity of data from IoT sensors to health servers that may get compromised by replay/MiTM attacks, or false data injection.

One of the revolutionary characteristics of blockchain technology is its potential to realize decentralized architecture for traditionally centralized services. In [37], authors develop a blockchain-based service for assigning software development tasks. Costumers publish tasks on the network, and are assumed to offer remunerations linearly correlated with the complexity of their tasks. Workers (service providers) are classified based on their available computing resources. A "matching smart contract" algorithm assigns tasks to workers with an equilibrium price, aiming to achieve market stability. While their system performs a one-to-one matching between multiple costumers and multiple workers, our verification scheme in this paper is primarily designed to facilitate one-to-many engagements, enabling a health authority (the HSP) to interact with multiple witnesses located in the vicinity of a target sensor. The HSP in our system architecture aims to choose the best combination of witnesses from which the highest verification probability is obtained; however, in [37] each costumer is connected to only a worker who is able to successfully complete the task.

In a slightly different context, authors of [38] employ blockchain to facilitate data sharing between different sensors of a smart transportation network. Their system enables connected vehicles (with embedded sensors) which are present in a suburb area to automatically sell their locally measured

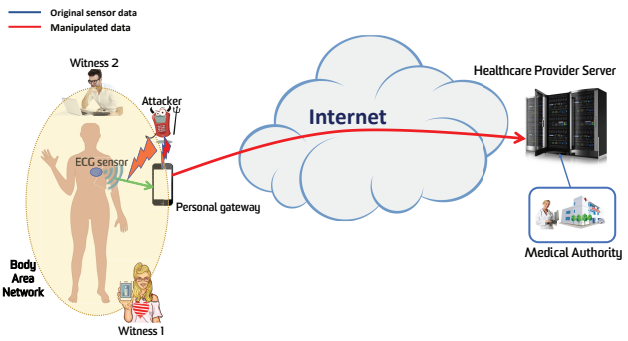


Fig. 1. Threat model: forging IoT data in transit by man-in-the-middle.

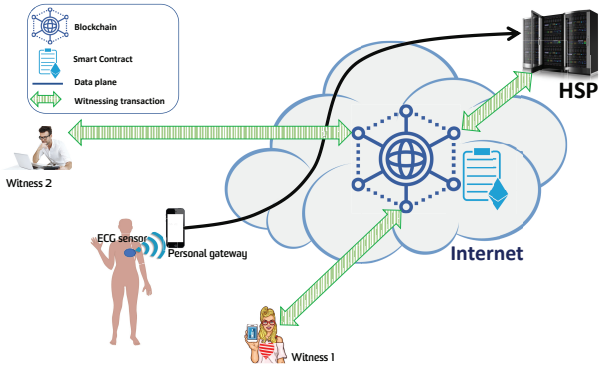


Fig. 2. System architecture of on-demand witnessing over blockchain

data (traffic load and weather condition) to other interested vehicles located in other parts of the city. While there are some overlaps between their system and this paper in terms of providing blockchain-based services for remote users, our verification scheme differentiates itself by following ways: (a) "use-case": our work is a security-oriented approach to detect a range of attacks on the integrity of sensitive and private data transmitted from a health sensor; however, their scheme in [38] facilitates trading of insensitive and public data between different sensors; (b) "infrastructure": while the private blockchain in [38] relies on a group of fixed pre-installed roadside units which enable the network connectivity for vehicles and their sensors. Our witnessing system, instead, can be implemented on conventional public blockchains without the need for building specialized infrastructure in the environment or having special hardware on witness devices; and (c) "buyer-seller interaction": their scheme connects multi-buyers to multi-sellers via an intermediate server. Buyers request data from this intermediate server that sends an announcement to fixed roadside units from which the connected sellers get notified of the request. The intermediate server runs an auction-based algorithm to connect buyers to best sellers. However, in our work the only buyer (the HSP) directly engages with potential sellers (witnesses), and selects the optimum combination of witnesses to maximize the probability of data verification probability.

III. System Architecture And Algorithm

In this section, we develop our system architecture for an on-demand auditing of wearable data using distributed witness statements that are enabled by smart contracts. We first outline the threat model, then develop our witnessing system architecture, and finally describe the flow of events in an operational scenario.

A. Threat Model

Body area networks (BAN) consist of wearable sensors that measure vital signs of body and transmit measurements toward a personal gateway at which the data is collected and forwarded to the HSP through the Internet (Fig. 1). In this paper, we focus on man-in-the-middle attackers that can manipulate IoT data in transit from wearable sensors to the personal gateway. The attacker could be either the owner of the sensor itself, falsifying the medical measurements in order to claim financial benefits, or other malicious entities who aim to harm to individual patients or the broader health-care system. In this environment: the attacker can compromise the gateway to forge the data measured and transmitted by the sensor; the attacker can backfill medical data [11]; and witnesses are independent and trusted entities which are assumed to be "not compromised". We would like to emphasize that assessing the trust level of individual witnesses is beyond the scope of this paper, and the only source of uncertainty comes from false positive rates of bloom filter used for generating witness statements (will be explained in §IV-A). Note that the HSP has the freedom to choose from a number of potential witnesses that are present in the environment.

B. System Architecture

Fig. 2 shows the system architecture of our witnessing scheme. We now explain various entities in this architecture.

Healthcare Service Provider (HSP) is an entity, shown on top right of Fig. 2, that provides remote health services to patients based on the medical data received from their body-worn sensor. The HSP receives, stores, and manages the access to real-time medical data to provide timely support and treatment to the patients.

Wearable IoT Sensor is an on-body low-power sensor to measure physiological signs of the patient. The sensor transmits the health data to a personal gateway from where the data gets forwarded to a remote server of the HSP on the Internet.

Witnesses, by their definition, are wireless nodes with Internet connectivity (*e.g.,* Witness1 and Witness2 in Fig. 2) that reside in the physical vicinity of the wearable IoT sensor, and share the same wireless broadcast domain. Witness devices (like smart phones, tablets, or even laptops) are expected to have sufficient computing capability to interact with a public blockchain, and are able to overhear the data transmitted wirelessly by the wearable sensor on the network. Since the wearable data is encrypted, witnesses are not able to extract any information from the overheard data packets.

Blockchain Network maintains a distributed public ledger, shared across peering nodes. Transactions between nodes get recorded in a chronological order, and are linked to previous ones by cryptographic mechanisms. All transactions within a specific time period are combined into a block which is

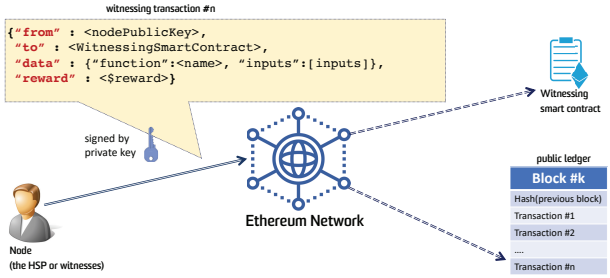


Fig. 3. Witnessing transaction sent by nodes on Ethereum blockchain, updating the state of smart contract and appending the ledger.

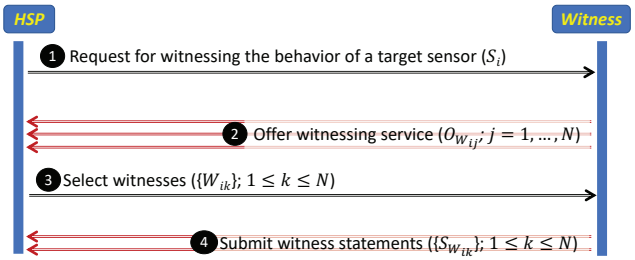


Fig. 4. Sequence of witnessing interactions between the HSP and potential witnesses over blockchain.

chained (linked) to previous blocks, extending the ledger. A new block can only be added to the ledger if it is approved by an internal "consensus" mechanism, *i.e.,* a majority of network nodes (aka *miners*) verify the validity of new blocks. Bitcoin and the initial version of Ethereum use Proof-of-Work (POW) for their consensus algorithm; however, Ethereum2 uses Proof-of-Stake (POS) which is a more energy-efficient method. Traditional blockchain networks like Bitcoin are typically used for financial transactions (exchange of digital money) [35], while modern networks like Ethereum serve for a wider range of customized and programmable transactions (exchange of valuable tokens based on smart contracts) [24]. Nodes (the HSP and witnesses) can join a blockchain network by creating their account, consisting of a pair of public and private keys – the private key is used to sign transactions, and the public (their identity across the network) will be used for verification.

Smart Contracts are special accounts that are created for specific application (*i.e.,* witnessing services) by a node (*i.e.,* the HSP in this paper), and become available to every nodes on the Ethereum network. Smart contracts come with a unique identifier, and typically offer a range of functions (§III-C) that can be called by a node which submits a transaction on the blockchain. The access to these functions can be controlled (*e.g.,* available to all nodes or restricted to certain nodes) by the node which develops and deploys the smart contract on the blockchain.

Our on-demand verification scheme is realized by executing certain functions of a witnessing smart contract on top of blockchain – it is in fact a distributed application (DApp) running over-the-top of blockchain, and hence independent of blockchain's internal processes (consensus included). Fig. 3 shows how a witnessing transaction is made on the Ethereum network, updating the state of the contract (depending on the specific function call) and ultimately getting appended to the ledger. Each transaction consists of four essential fields including: (a) `"from"` the identity of sender node, (b) `"to"` the identity of the witnessing smart contract, (c) `"data"` which contains of a `"function"` name along with a pair of `"inputs"` and `"outputs"` that vary by function (§III-C), and (d) `"reward"` offered by the caller to miners of the blockchain to approve the witnessing transaction and append it to the public ledger. We note that the HSP and potential witnesses on this network are particularly interested in the witnessing transactions (*i.e.,* addressed to the witnessing smart contract) of the ledger – other types of transaction may occur on this blockchain network.

C. Flow of Events

Wearable sensors transmit their measured data via their gateway to the HSP server on the Internet (shown by solid black line in Fig. 2). Note that the data is sent over the wireless medium on the local network (*e.g.,* inside hospitals or homes) to the gateway, and hence all communications between gateway and wearable healthcare devices are overheard by other parties in the vicinity. Man-in-the-middle attackers (with different incentives) can intercept packets sent by the wearable devices and may attempt to tamper the data in transit. There are multiple adversary models [39], [40] for manipulating data while being transmitted on the network: key management process (between the IoT device and HSP server) can be exploited by the man-in-the middle to access the secret key; an attacker may also collect previous packets of the IoT device and re-send them maliciously at later times to mount a replay attack on the HSP. For a given environment (say, a home), we have an IoT sensor (S_i) with a personal gateway (GW_i), and multiple witnesses (W_{ij}). Fig. 4 illustrates the flow of events in a witnessing process that consists of a sequence of four steps explained as follows.

Step1 **Request for Witnessing:** The HSP initiates the process by invoking a function (of the smart contract) called `"request"`, passing two parameters namely the identity of a target sensor (*e.g.,* `"device":<hash(mac)>`), and a desired duration of witnessing `"duration":<time>` for `"inputs"` component of the `"data"` field, as shown on the top left of Fig. 5. Note that for privacy reasons a hashed version of the sensor's MAC address (`"hash(mac)"`) is publicly announced on the network. Also, the `"request"` function can only be called by the HSP node, and potential witnesses will continuously look for this specific transaction submitted by the HSP. Upon arrival of a witnessing request, witness nodes will check their local network (wireless LAN) whether they are in the vicinity of the target sensor, or not. Note that the HSP would require witnesses to complete an eligibility challenge, proving that they are indeed in the vicinity of the target sensor location. The eligibility challenge has been widely studied in the literature [41], [42], [43], [44], [45], and is beyond the scope of this paper. That being said, wireless SSID association, public IP subnet, or *nonce* packets emitted by the target sensor can be used as a loose proof of location.

Step2 **Offer Witnessing Service:** Those nodes which find themselves local to the target sensor, may choose to offer the witnessing service depending upon their available resources

like memory, battery, or compute power. Each potential witness will respond by invoking a function of the smart contract called `"offer"`, passing a response to the `"eligibility"` challenge (*e.g.,* SSID), the `"granularity"` of their statements (*e.g.,* number of packets to be included per statement), the dollar `"cost"` of their service, and (optional) a `"deadline"` by which they aim to make their statement available on the blockchain. Statements can only get added to the ledger if they are approved by a majority of the blockchain miners. We note that miners prioritize those transactions (block of transactions) which offer higher rewards, and hence the approval of less-rewarding transactions can get delayed.

The granularity of witness statements depends on the resources such as memory, battery, and compute power that are available on the potential witness device for generating, transmitting, and more importantly submitting statements to the blockchain – the higher the granularity, the more resources needed, and thus resulting in higher price (cost) of witnessing. The deadline of statement depends on the load of the blockchain network and the dollar incentive (reward) that the sender (witness) is willing to incur. We note that a congested network may delay the availability of statements on the ledger, unless a higher reward is paid. The witnessing price depends on the deadline metric plus an additional remuneration that individual witnesses may want to receive from the HSP for the service to be provided. This means that availing a high granular statement in a short time from a demanding witness, can be quite expensive for the HSP.

Step3 **Witness Selection:** Once the witnessing offers are submitted by the eligible witnesses, the HSP needs to select those which give the best quality statements subject to a limited (dollar) budget available. Note that the HSP aims to achieve a verification probability close to 1 for the data transmitted by its target sensor – this objective is met when a large number of witnesses submit fine-grained statements. Considering the objective and constraint, the HSP selects a set of (zero, one or more) witnesses and announces them on the blockchain, committing to a dollar cost in exchange of the witnessing services to be provided by those selected witness nodes. This announcement will be made on the blockchain network by invoking `"select"` function of the witnessing smart contract. Again, the access to this function is restricted to the HSP as the authority in charge of witness selection.

Step4 **Submit Witness Statement:** Once a witness node is notified of its selection via the `"select"` transaction submitted by the HSP, it starts overhearing the packets transmitted by the target sensor and generates witness statements. Each of these statements is sent onto the Ethereum network by calling `"submit"` function of the smart contract. Once this specific typs of transaction is approved by the network miners, the smart contract will automatically pay off *Ethers* equal to the requested amount of `"price"` (in step3) to the witnesses from the HSP's account.

IV. Witness Statements and Optimal Witness Selection

Once the HSP ensures that potential witnesses are in the vicinity of its target sensor, its objective becomes to select a group of them that yield the best verification accuracy, given a limited budget. In this section we first describe the structure of witness statements, and then develop an optimization problem of selecting witnesses.

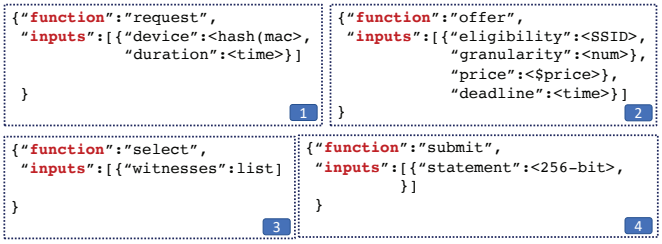


Fig. 5. Data field in witnessing transactions vary by function type.

A. Witness Statements and Verification

We consider a standard form of witness statements (with flexible granularity) that fulfill the following requirements: (a) not revealing any information of the sensor data, (b) allowing the HSP to check whether a specific data packet has been logged into the statement with some degree of certainty, and (c) being lightweight for resource-constrained witnesses to participate. One of the possible candidates for witness statement is bloom filter which was employed in [14] to provide opportunistic binding of the medical data to its context.

A bloom filter is a probabilistic data structure that is typically used to add elements (data packets) to a set (filter) and test if an element is in a set. Instead of the elements themselves, a hash of them is added to the set. When testing if an element is in the bloom filter, *false positives* are possible – either an element is definitely not in the set or that it is possible the element is in the set. An empty Bloom filter is a bit array of M bits, all set to 0. There are also k independent hash functions, each of which maps an element to one of the M bit positions. To add an element, feed it to the hash functions to get k bit positions, and set the bits at these positions to 1. Note that with more elements embedded in the filter, the error rate (false positive) increases. For given filter size M and the probability of false-positive f, the number of data elements n in the filter is determined by:

$$n = \frac{-M(ln2)^2}{ln(f)} \tag{1}$$

Also, the optimum number of hash functions needed to generate a bloom filter of size M bits with n logged elements is given by:

$$k = \frac{M}{n} ln2 \tag{2}$$

For our application, each witness commits to a number of inserted packets in each statement n (passed as `granularity` in step2) while the size of bloom filters for all witness statements is fixed (say, $M = 256$). Witnesses, therefore, may generate and submit a number (denoted by m) of statements based on their target false-positive probability f and the total number of packets (denoted by N) they have heard from the IoT sensor.

$$m = \frac{N}{n} = \frac{-Nln(f)}{M(ln(2))^2} \tag{3}$$

On the other hand, the HSP verifies the presence of certain packets in the submitted bloom filters by applying the same hash functions used to generate the statement. To test if a packet is in the filter, the HSP feeds it to the hash functions to get k bit positions. If any of the bits at these positions is 0, the packet definitely is not in the filter. If all are 1, then the packet may be in the filter with the probability of $1 - f$. It is important to note that a negative response from the statement is certain since bloom filter cannot result in a false negative. Since witnesses are independent, the verification probability (denoted by τ) using bloom filters submitted by W witnesses can be derived from:

$$\tau = 1 - \prod_{i=1}^{W} f_i \tag{4}$$

B. Optimal Witness Selection

As discussed in §III-C (step2) each potential witness offers a price for the service requested by the HSP. This price (denoted by α) is the sum of blockchain cost (paid to network miners for approving submitted statements) and the remuneration (reward) expected by potential witnesses. In absence of a behavioral model for the reward (requested by witnesses), we only consider the first component of the price, namely the blockchain cost of statement submission – we will derive (§V-A) the blockchain cost from our instance of private Ethereum network. A thorough study on the pricing model for individual rewards requested by witnesses is left for future work. Note that witness statements need to be approved by the blockchain miners, and thereby get appended as a valid transaction to the public ledger. Obviously, targeting a lower false-positive probability f requires a larger number of bloom filters (statements) to be submitted by the witness that results in a higher price charged to the HSP. Therefore, the cost of receiving witness statements from a witness which commits to error probability f at price α is given by:

$$c = m\alpha = \frac{-Nln(f)\alpha}{M(ln(2))^2} \tag{5}$$

Given a list of offers from W potential witnesses, each with (f_i, c_i), the HSP needs to select a combination of them that collectively give the highest verification probability (the lowest error) subject to its budget constraint (denoted by C). This can be formally defined as an optimization problem:

$$\begin{aligned} \max \quad & 1 - \prod_{i=1}^{W} (f_i)^{x_i} \\ \text{s.t.} \quad & \sum_{i=1}^{W} x_i c_i \leq C \end{aligned} \tag{6}$$

The objective function is maximized over x_i which indicates whether the i_{th} witness to be selected by the HSP.

$$x_i = \begin{cases} 1 & witness\ \boldsymbol{i}\ is\ selected, \\ 0 & witness\ \boldsymbol{i}\ is\ not\ selected, \end{cases} \tag{7}$$

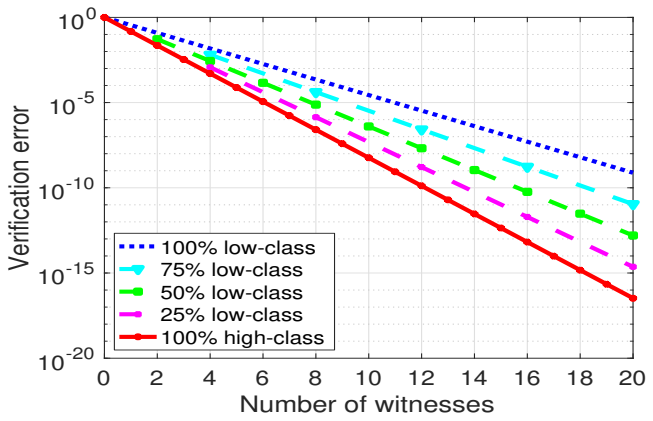
In order to better analyze this optimization problem, we consider two classes of the witnesses including: (a) "high-class" witnesses which are powerful devices, affording more memory and power for witnessing service (giving quality statements at higher price), and (b) "low-class" witnesses that are relatively low power (giving less-accurate statements at lower price). Let us assume there exist H high-class and L low-class potential witnesses, offering the pair of error and cost as (f_h, c_h) and (f_l, c_l), respectively. Now, our optimization problem becomes selecting the optimum number of witnesses from the two classes while keeping the total cost lower than a constant:

$$\begin{aligned} \min \quad & f_h^H f_l^L \\ \text{s.t.} \quad & c_h H + c_l L \leq C \\ & H, L \in Z^+ \end{aligned} \tag{8}$$

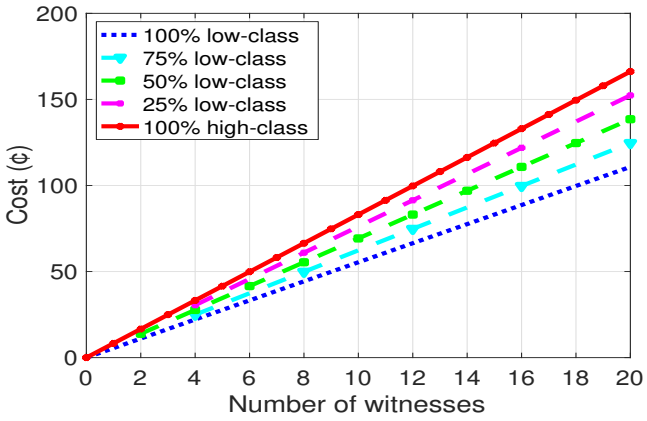
The optimization is performed over variables H and L, while false-positive probabilities (f_i's) and associated costs (c_i's) are known constants. We note that the objective function is a monotonically decreasing non-linear function of H and L. Intuitively, the verification error (our objective) decreases by the number of witnesses selected, resulting in higher costs. This trend is magnified by the quality of combined witnesses.

Fig. 6 shows the dynamics of verification error and total cost as a function of selected witnesses count – lines indicate the quality of selection (composition of high-/low-class witnesses). These values are computed by considering the following assumptions: $M = 256$ (size of bloom filter); $N = 150$ (total number of packets to be witnessed); $f_h = 0.15$ (false-positive rate of high-class witnesses); and $f_l = 0.35$ (false-positive rate of low-class witnesses). Given M, f_h and f_l, we compute $n_h = 64$ and $n_l = 117$ from Eq. 1, resulting the number of statements per witness $m_h = 3$ and $m_l = 2$. We note that Ether (ETH) is the fuel for an Ethereum network. In order to interact with the Ethereum blockchain, user nodes need to pay to miner nodes for the computation of that transaction. That payment is calculated in gas [46] , and gas is always paid in ETH. From our experimental setup (explained in §V-A), we found that submitting a 256-bit transaction requires spending $\alpha = 2.77¢$ [47]. Therefore, the price of offer for high-class and low-class witnesses, respectively, equals to $c_h = 8.31¢$ and $c_l = 5.54¢$ (from Eq. 5).

Here in Fig. 6, we consider five scenarios ranging from selecting purely low-class witnesses (shown by solid blue lines) to mixes of low-/high-class witnesses (shown by dashed lines) to purely high-class witnesses (shown by solid red lines). It can be seen that the verification probability (one minus error) and the total cost (for the HSP) monotonically increase by the number of witnesses. Also, improving the quality of witnesses from "100% low-class" to "100% high-class" accelerates the change rate (slope of lines) for both the verification error and the total cost. For example, given 10 witnesses, the verification error is 5×10^{-5} when all witnesses are chosen from the low-class type (solid blue line in Fig. 6(a)), resulting in a total cost



(a) Verification error.



(b) Total cost.

Fig. 6. Dynamics of (a) verification error, and (b) total cost, as functions of the number of selected witnesses and the quality of their statements.

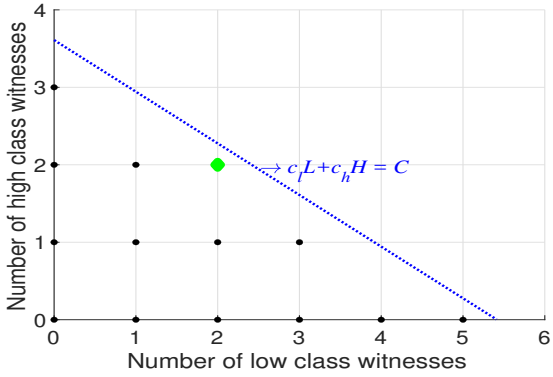


Fig. 7. Example of feasible region and optimal point of the integer linear programming for $C = 30¢$

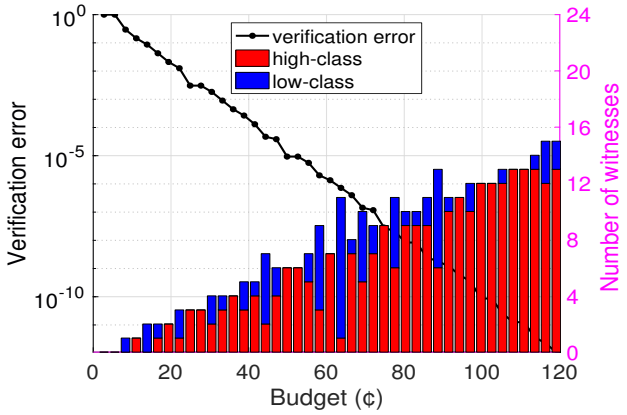


Fig. 8. Optimal verification error and composition of witnesses versus budget constraint.

of $20¢$ (solid blue line in Fig. 6(b)). Improving the quality of witnesses to high-class fraction being 50% (dashed green lines) and 100% (solid red lines) would reduce the error by 2 and 4 orders of magnitude, respectively, while incurring 25% and 50% additional cost.

We note that our optimization problem (8) is an integer non-linear programming. Since the objective function is a log-convex function and monotonically decreasing, its logarithmic transformation is also convex and monotonically decreasing [48]. This means that we can equally minimize the logarithmic transformation of the objective function in Eq. 8. Applying logarithmic transformation, multiplying the objective function by $\frac{-N\alpha}{M(ln(2))^2}$ and considering Eq. 5, our optimization problem is expressed as a standard integer linear programming (ILP):

$$\begin{aligned} \max \quad & c_h H + c_l L \\ \text{s.t.} \quad & c_h H + c_l L \leq C \\ & H, L \in Z^+ \end{aligned} \tag{9}$$

Restricting the variables to be positive ($H, L \in Z^+$) in conjunction with having only one linear constraint with a negative slope will result in a triangle feasible region. Now, problem (9) is a standard ILP, which does not have a closed-form solution [49]. We illustrate in Fig. 7 the feasible region with a cost constraint $C = 30$. The dotted blue line is our objective function and our optimal solution is the closest point (of the grid) to this line from the triangle region below it. In other words, the optimal solution is the combination of witnesses that yield a cost value close to this upper bound line. As highlighted by the green dot in Fig. 7, our optimal solution is obtained by selecting two high-class witnesses ($H = 2$) and two low-class witnesses ($L = 2$).

Fig. 8 shows the output of the optimization problem (Eq. 8) when the budget constraint varies from 0 to 120 cents. Obviously, for a budget constraint less than $c_l = 5.54¢$, it becomes infeasible to find the optimal solution (no witness can be selected). At a microscopic level, we observe that adding to the budget may at least increase the witnesses count or their quality. At a macroscopic level, instead, relaxing the budget constraint would result in a larger number of witnesses (height of stacked bars) along with an improvement in their quality (height of red bars).

V. Evaluation Results

We now evaluate the efficacy of our solution by applying it to real trace data. We first set up a private Ethereum environment to deploy our smart contract and obtain the cost (price) of submitting transactions on the blockchain network. As discussed in §IV-B, this price is considered as the sole contributor to the cost of selecting individual witnesses, and modeling witness rewards is beyond the scope of this paper. We next simulate our witness selection algorithm on trace data. Though our algorithm is designed for sensor networks at

home or hospital premises, obtaining WiFi data from sufficient households to test the algorithm at scale is very challenging. To validate our algorithm at larger scale, we use traces taken from the WiFi network of a multi-story building on our university campus.

A. Experiments with Ethereum Platform

We set up a private blockchain network on a machine (with 2.5 GHz Quad-Core Intel Core i7 and 6GB of memory) using a "private" Ethereum network [50]. The private Ethereum is available for research purposes and private business use-cases that allows developers to run their smart contracts on an isolated local Ethereum environment without the need to necessarily spend real ethers. All the nodes and transactions of a private network can be readily realized on the public network too, with no modification [51]. To interact with the blockchain network, we use the Go-version Ethereum client (Geth v1.9.7) [50]. Also, we develop our smart contract as an app with four functions (§III-C) in Solidity v0.5.15. Lastly, we deploy the smart contract on the Ethereum blockchain using the Ethereum JavaScript API called `web3.js` v1.2.6 [52]. Our experimental codes are publicly available on [53].

Each of the witnesses in our scenario can choose to individually participate in the witnessing service by responding to a request from the HSP, submitting their statements on the blockchain. Note that the cost of this service merely depends on the size of statement submitted by each witness, independent of other witnesses. To obtain the unit cost, we conducted an experiment on our private instance of Ethereum network (running on our local machine) by creating two representative nodes (a witness and an HSP). The experiment initiates by a transaction from the HSP, sending the witnessing smart contract (discussed in §III-C) to the Ethereum network. Once this transaction is fully executed (1,500,000 gas), the address of the smart contract becomes available to both HSP and witness. With known address of the contract, the HSP invokes the function "request" (300,000 gas) in Fig. 5. Thereafter, the witness invokes "offer" (23,000 gas) which is followed by a "select" call from the HSP, accepting the offer and authorizing the witness to access the function "submit". Finally, the witness submits its statement, embedded as input part of the function "submit" (each call 23,000 gas) and gets paid automatically from the HSP's account.

From our Ethereum testbed, we found that it requires $23,000$ gas in order to `submit` a witness statement of size 256 bits to the blockchain. The dollar value of each gas fluctuates based on the dynamics of the crypto-currency market. At the time of our experiments (25 January 2020), 1,000,000 gas = 0.0075 Ethereum (ETH) and 1 ETH = 160.36$ – this means that 23000 gas is equal to $\alpha = 2.77¢$ [47]. Also, the initial deployment of witnessing smart contract would cost 1,500,000 gas that is equal to 180¢ (one-off payment made by the HSP). Note that witnesses will pay an amount (proportional to the size of transactions) for all transactions including `offer` and `submit`. Since the `offer` transaction occurs once per each session of witnessing we only consider the cost of `submit` transactions in our witness selection algorithm.

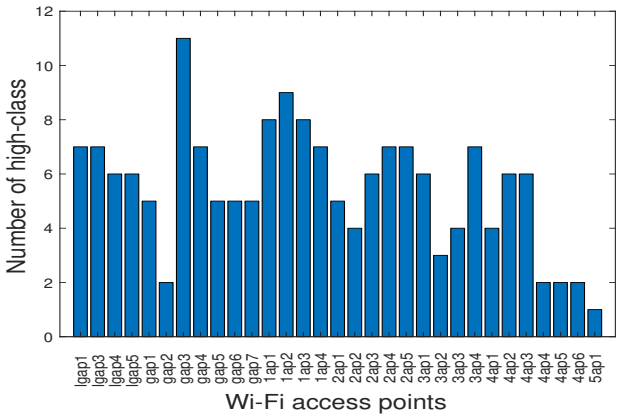


Fig. 9. Number of high-class witnesses available around each of 31 WiFi access points.

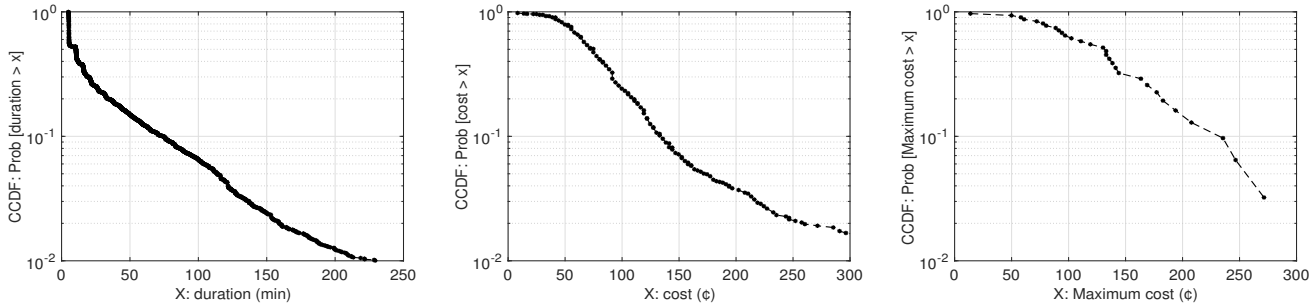
B. Evaluation in a Multi-Story Building

To evaluate the efficacy of our scheme, we obtained WiFi trace data from our University IT department. The data contains session logs during a day (12:00am-11:59pm) for 31 WiFi access points (AP) located in a 5-story building. Each record contains device MAC address (note that we have hashed this to preserve anonymity); a unique AP name that clearly indicates the building name, floor level, and access point ID; time at which the device associated to/disassociated from the AP (note that this is in minutes and therefore we do not have sub-minute accuracy); and avg throughput indicating data rate during the session.

To simulate our scheme, we assume that each AP (static and powerful) represents a high-class witness and a user device (mobile with limited compute power) represents a low-class witness. Since APs are spread across the building, we call the coverage area of each AP a "zone" from which user devices connect to the AP. Obviously wireless zones overlap, and hence for a given AP there exist a number of "neighbor APs" within its close proximity. A witnessing scenario in this environment is as follow. We consider our target sensor in an AP zone with several high-class witnesses (neighbor APs) and a number of low-class witnesses (user devices connected to the AP). Fig. 9 shows the distribution of high-class witnesses across 31 zones – each zone corresponds to a WiFi AP in the building. It can be seen that each zone comprises 5 high-class witnesses on average. This number varies in certain zones – for example zone *gap3* located at ground level covers 11 APs, or zone *5ap1* located at level5 accommodates only one AP.

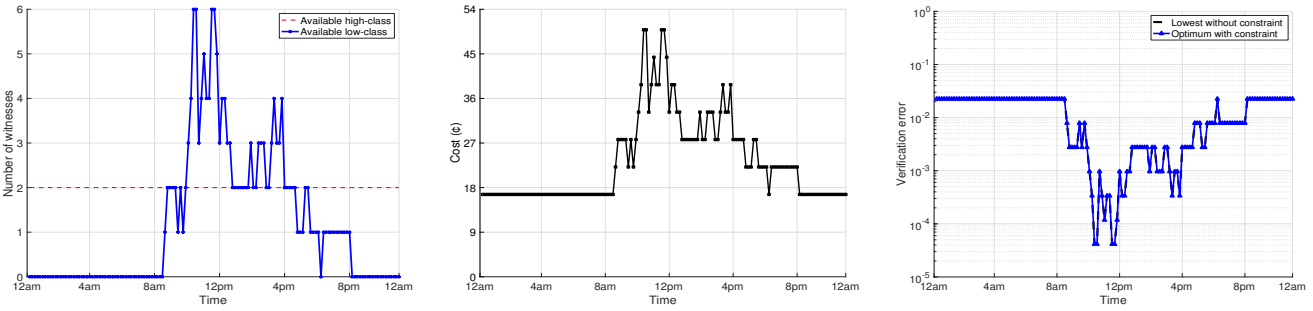
Obviously, high-class witnesses (neighbor APs) consistently stay present within their corresponding zones, but low-class witnesses (user devices) are mobile and hence their availability changes over time. To get a sense of the availability of low-class witnesses, we plot in Fig. 10(a) the complementary cumulative distribution function (CCDF) of all sessions duration in our trace. Of a total of 8263 WiFi sessions, more than half (52%) had a duration more than 10 minutes. Therefore, we consider a witnessing epoch to last 10 minutes. A target wearable sensor is assumed [14] to transmit (on average) 15 packets per minute, and hence a total of $N = 150$ packets will be witnessed during a witnessing epoch.

In our simulation, we divide a day into 144 distinct epochs



(a) Duration of WiFi sessions. (b) Cost of selecting all available witnesses. (c) Maximum epoch cost per zone.

Fig. 10. CCDF of: (a) duration of WiFi sessions in our trace data, (b) cost of selecting "all" available witnesses across all epochs and WiFi zones, and (c) maximum of epoch cost per zone.



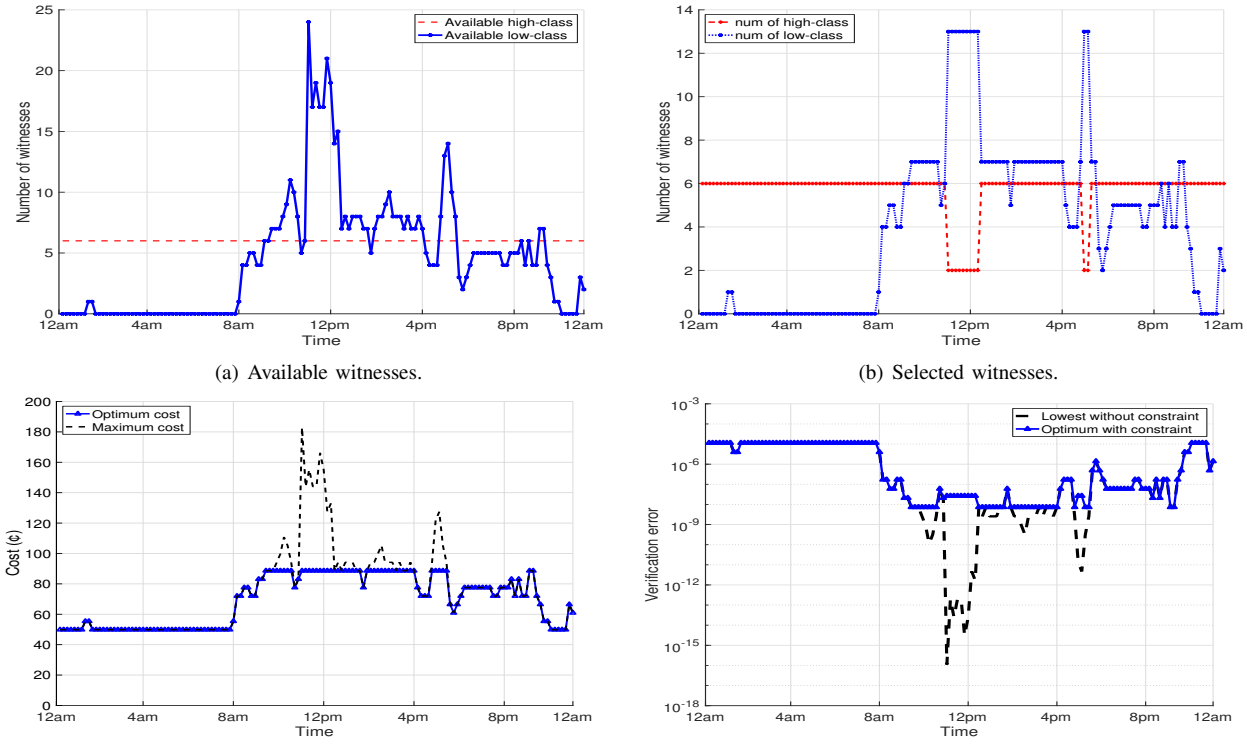
(a) Available witnesses. (b) Cost of witnessing. (c) Verification error of witnessing.

Fig. 11. Dynamics of witnessing for a low-density zone *4ap5* when optimization is not needed: (a) count of available witnesses, (b)) cost of witnessing, and (c) verification error of witnessing.

of 10-minute. For each epoch, we only consider those user devices as potential low-class witnesses that remain connected to the same zone for the whole duration of that epoch – devices which switch their zone during the epoch are filtered out. This means that potential witnesses are required to be in the vicinity of the target sensor (present in the zone) during an epoch. With this condition, 27% of sessions on average are removed per epoch-zone – note that in three-quarters of epoch-zones less than 40% of sessions are filtered out, while for a third of epoch-zones all sessions persist during the epoch (no session was filtered out).

Moreover, we note that cost constraints and pricing strategy chosen by the HSP can be influenced by a number factors such as the importance of data transmitted by the target sensor, the matter of urgency for detecting security breaches, the availability of potential witnesses, or the number of sensors to be witnessed at scale. It is important to note that having a generous budget for the HSP may result in selection of all available witnesses, making the optimization unnecessary. On the other hand, a very tight budget can make it infeasible for the optimization algorithm to select even one low-class witness. For this paper, we consider a simple pricing strategy for the HSP whereby a fixed budget is pre-decided for every epoch, and all epochs are treated equally – more sophisticated strategies can explored in future works.

Recall from §IV-A, selecting a witness would cost $c_h = 8.31¢$ for the high-class and $c_l = 5.54¢$ for the low-class, given $\alpha = 2.77¢$ (computed from our Ethereum testbed in §V-A) and $N = 150$ packets transmitted by the target sensor over a 10-minute epoch. Note that the number of witnesses (both high-class and low-class) during each epoch is known, and hence the incurred cost incurred of selecting all available witnesses (in a given zone) can be computed. Fig. 10(b) shows the CCDF of cost per epoch during day-time (*i.e.,* 8am-5pm when user devices are likely present) across all zones. We observe that for a given zone/epoch there is an 80% chance to have the cost more than 50¢, when the HSP chooses to select all available witnesses in the sensor environment. Note that some zones get more crowded than others depending on their location (meeting rooms, study spaces, offices, research labs, or entry/exit points) in the building. We computed the maximum cost of each zone across epochs of the day. Fig. 10(c) shows the CCDF of maximum cost per zone. We observe that for a third of zones (low-density ones) the cost is always below 90¢, hence for these zones the HSP may prefer to go with all available witnesses in the environment without employing the optimization algorithm. We, therefore, set the budget constraint to a fixed value equals to 90¢ in our simulations, highlighting the fact that the optimization may be needed in two-third of the zones (high-density ones) where there likely be a large number of mobile devices available to perform witnessing. Note that our "static" budget constraint is chosen based on empirical observations from a representative environment, purely for demonstrating the feasibility of a single epoch witnessing via blockchain logging contract. Given mobile nature of wireless nodes, a successful witnessing can only be achieved during short time intervals, and hence witnessing longer than a few minutes requires a dynamic strategy in selecting witness nodes over multiple epochs. This would entail a dynamic pricing strategy for a multi-epoch witnessing which is beyond the scope of this paper.



(a) Available witnesses.

(b) Selected witnesses.

(c) Cost of witnessing.

(d) Verification error of witnessing.

Fig. 12. Dynamics of witnessing for a high-density zone *4ap2* when optimization is needed: (a) count of available witnesses, (b) count of selected witnesses, (c) cost of witnessing, and (d) verification error of witnessing.

Given the budget constraint, we now simulate our scheme during the whole 24 hours in two representative zones: a low-density (AP *4ap5*) and a high-density (AP *4ap2*). Fig. 11 shows the dynamics of witnessing (number of witnesses, cost, and verification error) for the low-density zone. It is seen in Fig. 11(a) that this zone accommodates two high-class witnesses (shown by a flat dashed red line) and up to 6 low-class witnesses (mostly 1-3 during working hours as shown by dotted blue lines). For this low-density environment, no optimization is needed since the total cost of selecting all available witnesses is well below the budget constraint 90¢, and hence all available witnesses are selected. We observe in Fig. 11(b) that the highest cost is about 54¢ for epochs between 10am-11:30am when the total count of witnesses reaches to its maximum of 8. As a result, the lowest verification error becomes identical to the optimal verification, as shown in Fig. 11(c) – the lowest error of 10^{-4} can be achieved at the cost of 54¢ per epoch during the peak time of this zone.

Moving to the high-density zone which hosts 6 high-class witnesses and 5-24 low-class witnesses during day time, as shown in Fig. 12(a). The HSP now needs to run the optimization algorithm for selecting the optimal combination of witnesses available in the environment. We observe that during early morning (12am-9am), selecting all available witnesses results a cost less than the constraint 90¢, as shown in Fig. 12(c). Following 9am, it is seen that the cost is saturated at 90¢ as a result of optimization. Focusing on optimization results, we observe that two-third of high-class witnesses are left out due to abundance of low-class witnesses during busy epochs (around 12pm and closer to 5pm), as shown in Fig. 12(b). In this scenario, the optimum verification error would be higher than the lowest error as shown in Fig. 12(d) – during the peak time the best error 10^{-8} is achieved by selecting 2 high-class and 13 low-class witnesses.

VI. Conclusion

In this paper, we have proposed an on-demand and distributed scheme to verify healthcare IoT data. Our architecture provides the motivation and means to engage via smart contracts on a blockchain: Health authorities can request witness statements needed for data verification of target sensors; local witness devices can monetize their statements without compromising their privacy. We developed an optimization algorithm for health authorities to select an optimal collection of available witnesses to achieve the best verification probability subject to a budget constraint. We simulated our algorithm on real data captured from WiFi connections in a multi-story campus building to show that a verification probability of more than 99% can be achieved at cost of less than two dollars for one-hour witnessing service. This work is the first step towards on-demand witnessing of sensors network data, applicable to real-world scenarios. In future work we plan to develop a method to dynamically adjust the budget constraint for improving the verification probability over longer periods with variable number of witnesses. We will also consider a more accurate model for overhearing in lossy wireless networks.

References

[1] 10 Ways The Internet of Medical Things Is Revolutionizing Senior Care. [Online]. Available: http://bit.ly/2N7Q1Kz

[2] Internet of Things (IoT) Healthcare Market is Expected to Reach $136.8 Billion Worldwide, by 2021 - MarketWatch. [Online]. Available: https://on.mktw.net/2QgDrq6

[3] B. Celler, M. Varnfield, R. Sparks, J. Li, S. Nepal, J. Jang-Jaccard, S. McBride, and R. Jayasena, "Home monitoring of chronic disease for aged care," *Australian e-Health Research Centre (AEHRC) CSIRO*, 2016.

[4] Oscar Health Using Misfit Wearables To Reward Fit Customers. [Online]. Available: http://bit.ly/2N4xFKk

[5] Beyond Data: Are connected Medical Equipmentand Werables the Next Big Target for Cyberattack? [Online]. Available: https://goo.gl/kzCTG2

[6] Connected Device System Validation & Quality - Best Practices. [Online]. Available: https://goo.gl/68d7ng

[7] Cyber Security: the Risk Posed by Smart Medical Devices. [Online]. Available: https://goo.gl/Z3XNPT

[8] M. Ammar, G. Russello, and B. Crispo, "Internet of things: A survey on the security of iot frameworks," *Journal of Information Security and Applications*, vol. 38, pp. 8–27, 2018.

[9] V. Hassija, V. Chamola, V. Saxena, D. Jain, P. Goyal, and B. Sikdar, "A survey on iot security: application areas, security threats, and solution architectures," *IEEE Access*, vol. 7, pp. 82 721–82 743, 2019.

[10] B. K. Mohanta, D. Jena, S. Ramasubbareddy, M. Daneshmand, and A. H. Gandomi, "Addressing security and privacy issues of IoT using blockchain technology," *IEEE Internet of Things Journal*, 2020.

[11] M. Siddiqi, V. Sivaraman, and S. Jha, "Timestamp Integrity in Wearable Healthcare Devices," in *Proc. IEEE ANTS*, Bangalore, India, Nov 2016.

[12] Hackers Can Remotely Access Syringe Infusion Pumps to Deliver Fatal Overdoses. [Online]. Available: https://bit.ly/2RI4LR8

[13] 2015 State of the Smart Home Report . [Online]. Available: https://bit.ly/2ZdY6kE

[14] M. Siddiqi, S. T. Ali, and V. Sivaraman, "Secure Opportunistic Contextual Logging for Wearable Healthcare Sensing Devices," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[15] ——, "Forensic verification of health data from wearable devices using anonymous witnesses," *IEEE Internet of Things Journal*, 2020.

[16] B. Farahani, F. Firouzi, V. Chang, M. Badaroglu, N. Constant, and K. Mankodiya, "Towards fog-driven iot ehealth: Promises and challenges of iot in medicine and healthcare," *Future Generation Computer Systems*, vol. 78, pp. 659–676, 2018.

[17] Telehealth could save $3 billion a year to health budget. [Online]. Available: https://bit.ly/32UYeH1

[18] Examining the Costs of Telehealth. [Online]. Available: https://bit.ly/2QQYNvG

[19] Cost savings from a telemedicine model of care in northern Queensland, Australia. [Online]. Available: https://bit.ly/3bkLxJn

[20] John Hancock Introduces a Whole New Approach to Life Insurance in the U.S. That Rewards Customers for Healthy Living. [Online]. Available: https://prn.to/2ESEvQd

[21] UnitedHealthcare and Qualcomm Collaborate to Launch New Wellness Program That Links Financial Incentives with the Use of Wearable Devices. [Online]. Available: https://bit.ly/351XDGe

[22] R. Accorsi, "Safe-Keeping Digital Evidence with Secure Logging Protocols: State of the Art and Challenges," in *Proc. IEEE IT Security Incident Management and IT Forensics (IMF)*, Stuttgart, Germany, Sep 2009.

[23] M. Siddiqi, S. T. All, and V. Sivaraman, "Secure lightweight context-driven data logging for bodyworn sensing devices," in *Proc. IEEE Digital Forensic and Security (ISDFS)*, Tirgu Mures, Romania, Apr 2017.

[24] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *International Journal of Web and Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.

[25] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on software engineering*, no. 2, pp. 222–232, 1987.

[26] A. Hamza, H. Habibi Gharakheili, T. A. Benson, and V. Sivaraman, "Detecting Volumetric Attacks on LoT Devices via SDN-Based Monitoring of MUD Activity," in *Proc. ACM SOSR*, San Jose, CA, USA, Apr 2019.

[27] A. Sivanathan, H. Habibi Gharakheili, and V. Sivaraman, "Detecting Behavioral Change of IoT Devices using Clustering-Based Network Traffic Modeling," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7295–7309, Aug 2020.

[28] A. Hamza, H. Habibi Gharakheili, and V. Sivaraman, "Combining MUD Policies with SDN for IoT Intrusion Detection," in *Proc. ACM IoT Security and Privacy*, Budapest, Hungary, Aug 2018.

[29] B. Schneier and J. Kelsey, "Secure audit logs to support computer forensics," *ACM Transactions on Information and System Security (TISSEC)*, vol. 2, no. 2, pp. 159–176, 1999.

[30] J. E. Holt, "Logcrypt: forward security and public verification for secure audit logs," in *Proceedings of the 2006 Australasian workshops on Grid computing and e-research-Volume 54*. Australian Computer Society, Inc., 2006, pp. 203–211.

[31] D. Ma and G. Tsudik, "A new approach to secure logging," *ACM Transactions on Storage (TOS)*, vol. 5, no. 1, p. 2, 2009.

[32] Z. Zhu and G. Cao, "Applaus: A privacy-preserving location proof updating system for location-based services," in *INFOCOM, 2011 Proceedings IEEE*. Citeseer, 2011, pp. 1889–1897.

[33] X. Wang, J. Zhu, A. Pande, A. Raghuramu, P. Mohapatra, T. Abdelzaher, and R. Ganti, "Stamp: Ad hoc spatial-temporal provenance assurance for mobile users," in *2013 21st IEEE International Conference on Network Protocols (ICNP)*. IEEE, 2013, pp. 1–10.

[34] R. Hasan, R. Khan, S. Zawoad, and M. M. Haque, "Woral: A witness oriented secure location provenance framework for mobile devices," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 1, pp. 128–141, 2016.

[35] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 2084–2123, 2016.

[36] A. Ouaddah, A. Abou Elkalam, and A. Ait Ouahman, "Fairaccess: a new blockchain-based access control framework for the internet of things," *Security and Communication Networks*, vol. 9, no. 18, pp. 5943–5964, 2016.

[37] W. Zhang, Z. Hong, and W. Chen, "Hierarchical pricing mechanism with financial stability for decentralized crowdsourcing: A smart contract approach," *IEEE Internet of Things Journal*, 2020.

[38] W. Chen, Y. Chen, X. Chen, and Z. Zheng, "Toward secure data sharing for the iov: a quality-driven incentive mechanism with on-chain and off-chain guarantees," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 1625–1640, 2019.

[39] I. Andrea, C. Chrysostomou, and G. Hadjichristofi, "Internet of things: Security vulnerabilities and challenges," in *2015 IEEE Symposium on Computers and Communication (ISCC)*. IEEE, 2015, pp. 180–187.

[40] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, "A survey on security and privacy issues in internet-of-things," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1250–1258, 2017.

[41] J. Hightower, R. Want, and G. Borriello, "Spoton: An indoor 3d location sensing technology based on rf signal strength," 2000.

[42] L. Lazos, R. Poovendran, and S. Čapkun, "Rope: robust position estimation in wireless sensor networks," in *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, 2005, p. 43.

[43] Z. Li, W. Trappe, Y. Zhang, and B. Nath, "Robust statistical methods for securing wireless localization in sensor networks," in *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, 2005, p. 12.

[44] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust distributed network localization with noisy range measurements," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*. ACM, 2004, pp. 50–61.

[45] S. Capkun and J.-P. Hubaux, "Secure positioning of wireless devices with application to sensor networks," in *IEEE infocom*, no. CONF, 2005.

[46] Gas (Ethereum). [Online]. Available: https://www.investopedia.com/terms/g/gas-ethereum.asp

[47] Ethereum Gas Tracker — Etherscan. [Online]. Available: https://bit.ly/30bJOm2

[48] D. G. Luenberger, Y. Ye *et al.*, *Linear and nonlinear programming*. Springer, 1984, vol. 2.

[49] S. P. Bradley, A. C. Hax, and T. L. Magnanti, "Applied mathematical programming," 1977.

[50] Go Ethereum. [Online]. Available: https://github.com/ethereum/go-ethereum

[51] Ethereum Private Network – Create your own Ethereum Blockchain! [Online]. Available: https://bit.ly/3dRmjm7

[52] web3.js - Ethereum JavaScript API. [Online]. Available: https://bit.ly/3f3OY7M

[53] M. H. Chinaei. (2020) Witnessing Smart Contract. [Online]. Available: https://github.com/MohammadHosseinChinaei/Blockchain-based-Witnessing

Mohamamd Hossein Chinaei received his B.Sc. and M.Sc. degrees of Electrical Engineering from Isfahan University of Technology, Isfahan, Iran in 2011 and 2014 respectively, and is currently a Ph.D. candidate at the University of New South Wales in Sydney, Australia. His research interests include distributed ledger technology, network security, and Internet of Things.

Hassan Habibi Gharakheili received his B.Sc. and M.Sc. degrees of Electrical Engineering from the Sharif University of Technology in Tehran, Iran in 2001 and 2004 respectively, and his Ph.D. in Electrical Engineering and Telecommunications from UNSW in Sydney, Australia in 2015. He is currently a Senior Lecturer at UNSW Sydney. His current research interests include programmable networks, learning-based networked systems, and data analytics in computer systems.

Vijay Sivaraman received his B. Tech. from the Indian Institute of Technology in Delhi, India, in 1994, his M.S. from North Carolina State University in 1996, and his Ph.D. from the University of California at Los Angeles in 2000. He has worked at Bell-Labs as a student Fellow, in a silicon valley start-up manufacturing optical switch-routers, and as a Senior Research Engineer at the CSIRO in Australia. He is now a Professor at the University of New South Wales in Sydney, Australia. His research interests include Software Defined Networking, network architectures, and cyber-security particularly for IoT networks.