Towards Blockchain for Decentralized Self-Organizing Wireless Networks

Steven Platt

*Department of Information and Communication Technologies
Engineering
Universitat Pompeu Fabra
Barcelona, Spain
steven.platt@upf.edu*

Miquel Oliver Riera

*Department of Information and Communication Technologies
Engineering
Universitat Pompeu Fabra
Barcelona, Spain
miquel.oliver@upf.edu*

Abstract— Distributed consensus mechanisms have been widely researched and made popular with a number of blockchain-based token applications, such as Bitcoin, and Ethereum. Although these general-purpose platforms have matured for scale and security, they are designed for human incentive and continue to require currency reward and contract functions that are not requisite in machine communications. Redes Chain is a new blockchain, built to support fully decentralized self-organization in wireless networks - without a cryptocurrency or contract dependency.

Keywords— *Blockchain, Self-Organizing Networks, Distributed Ledger Technologies, 802.11, 5G, OpenWRT, Spectrum Sharing*

I. INTRODUCTION

Initially popularized through application in digital currency, distributed ledger technologies (DLT) are now seeing wider adoption as a path for extending peer-to-peer design and security to the broader internet. To allow open participation, a number of DLT designs deploy computationally expensive cryptography paired with digital currency reward [1], creating a format optimized for human incentive and trust that is a less natural fit for machines. Although popular applications of blockchain including Bitcoin and Ethereum, tie blockchain to a digital currency function - it is important to note that blockchain as a data structure has no native association to digital currency. It is this isolated application of the blockchain data structure, and its ability to support distributed consensus, that is the focus of this research.

The ability to form consensus among equal peers has a number of implications in networks research - but we propose its most natural application is that of decentralizing self-organization functions in wireless networks - systems which by design are dependent on coordination and context sharing among network participants.

In the following sections of this paper, we detail the use cases and limitations of existing cryptocurrency-based blockchain ecosystems Bitcoin and Ethereum. We then define the broader Self Organizing Network use case - presenting a new model, built atop blockchain, that is wholly decentralized. Next, we present Redes Chain - a new blockchain prototype, designed for machine communications and to allow the decentralization of self-organization functions in wireless networks. Finally, the Redes blockchain prototype is demonstrated through a proof-of-concept deployment

handling access federation among independent 802.11 networks.

II. BITCOIN AND ETHEREUM: BLOCKCHAIN FOR CURRENCY AND CONTRACTS

Today Bitcoin is the most well-known application of blockchain technology, but also the oldest and simplest technical implementation in popular use. The original Bitcoin whitepaper was published in 2008, and detailed the design of a digital currency system that removed the need for a trusted third party for the verification of transactions [1]. In being designed as a digital currency, the Bitcoin ledger structure can be simplified as a state transition system; with the current state represented as the total ownership of all digital coins at a moment in time, and the state transitions represented by the movement of these coins, or payments made between users in the network.

To aid in decentralization, Bitcoin is designed as a permissionless network, allowing nodes or participants to join and leave the network at any time - with all transaction data sent as broadcast. After data is broadcast, computers in the network compete to find a hash of the block data that is smaller than a threshold size or difficulty set for the entire network. The hash difficulty of the bitcoin network is a composite function of its block throughput target and the combined hash power of the total network (1). At the time of writing, hash difficulty is adjusted so as to keep block creation constant at roughly one block every ten minutes [2].

$$\text{hash difficulty} = f(\text{block throughput target}(\text{network hash power})) \quad (1)$$

Because the result of hashing is pseudo-random, it is believed that every computer in the network of equal computing power, has an equal chance of being first to find the correct hash [3]. The equality created through the pseudo-random hash function also makes the network able to remain secure as long as a simple majority, or 51% of the network are acting in good faith, but results in a number of required hashes that is a further composite function of the previously mentioned hash difficulty, and the number of pending blocks being added in the network (2).

This work was partially supported by the Spanish and Catalan Governments through the project "Plan Nacional": AEI/FEDER TEC2016-79510 "Redes Con Celdas Densas y Masivas".

$$\text{required hashes} = f(\text{hash difficulty}(\text{pending blocks})) \quad (2)$$

To provide incentive for doing the difficult computation work, machines participating in the network are issued a reward in the form of Bitcoin, for finding and broadcasting the first hash successfully. These reward payments are covered by transaction fees charged to users wishing to add blocks to Bitcoin's chain. Because this hash value can be verified by others in the network - this process of consensus is named "Proof of Work" (POW). A primary side effect of the race condition created through POW consensus, is that power used for all unsuccessful hashes is considered wasted, making the system highly resource inefficient. A secondary behavior and weakness of using the POW consensus model is that it makes financial incentive in the form of block rewards and transaction fees, native to the operation of the system.

In 2013, Vitalik Buterin published the Ethereum whitepaper, seeking to expand the functions of Bitcoin - into a general purpose compute platform. The Ethereum blockchain included a more complex block structure that allowed storing logic which executes only when preset conditions are met. This new block structure allowed the creation of contracts, but retained Bitcoin's permissionless format, POW consensus, and block rewards [3]. Programmed into the contract support of Ethereum, is the ability to create a

development of the LTE cellular standard. SON functions today are formed in one of three designs: centralized, distributed, and hybrid [6]. In a centralized design, resource management and air interface coordination algorithms are processed by a central controller. With a distributed model, these algorithms are run at the network edge. Finally, a hybrid model employs a combination of the former two [6]. It is important to note that although a distributed model allows algorithms to run at the network edge, these controls remain limited to coordination within a single operator environment, often relying on S2 interface connections to a carrier core in cellular deployment, or a hub controller in 802.11x networks for compatible hardware actuation and control [7].

With a target to reduce manual administration by automating routine configurations in cellular networks; the SON standards developed by the 3GPP eventually included provisions for energy savings, handover optimization, automatic neighbor relation management, and load balancing [7]. Today these cellular-centric SON operations have also been extended to Wi-Fi and other air interface networks which benefit from the enhanced environment knowledge and distributed coordination capability that SON provides [8]. The Redes blockchain proof of concept, presents a new fourth model of wireless SON functionality, built in a fully decentralized context, allowing wider coordination among isolated networks that do not share controller or S2 interface connectivity (fig. 1).

Fig. 1. SON Architectures

secondary digital currency token, pegged to the value of Ethereum's own digital currency, Ether. These tokens are referred to as ERC-20, and in effect, allow a white labelling of the core Ether token, while retaining compatibility with the broader ecosystem of Ethereum smart contracts [4]. A number of network systems have been built on top of the Ethereum blockchain, but in doing so, these systems must inherit Ethereum's contract structure, with peer-to-peer payments at the core. An example of such system is Privatix Network, a VPN service allowing peer-to-peer payment for bandwidth used while hosting VPN connections [5].

III. SELF-ORGANIZING NETWORKS

To address concerns of increasing complexity in cellular networks, the 3GPP completed work to formally define Self Organizing Network (SON) functions in tandem with the

IV. REDES: BLOCKCHAIN FOR DECENTRALIZED SELF-ORGANIZING WIRELESS NETWORKS

The Redes blockchain is structured as a permissioned chain and does not attempt to enforce code execution commitments in the form of smart contracts as with Ethereum. The Redes proof of concept as presented, deploys blockchain in its more basic form - as a decentralized data store for network specific state data. With Redes, data storage and hardware specific actuation and control operate separately (fig. 2). Structuring the chain in this way allows the benefits of developing distributed consensus while leaving code execution control with individual network operators who can optimize for various combinations of hardware in their environment. The Redes blockchain is unique in that it assumes inherent value for the data itself, demonstrated

through examples such as spectrum sharing facilities, where a wider environment context is required for optimal network function [9].

Fig. 2. Redes separation of consensus and application code

A. System Requirements

The Redes blockchain is written in the Python programming language, for hardware isolation, and portability across infrastructure. Using Python version three for the construction of Redes also allowed the use of Python libraries and micro-frameworks - including Flask and SQLite for full web server and database functions in a compact package suitable for embedded systems use. Since it is not intended as a one-size application platform, or a digital currency, Redes can strip out dependencies that would require a full Linux operating system or more robust database systems as seen in larger Ethereum, and Bitcoin derived projects [4]. In current form, the Redes Blockchain code utilizing less than 20kb of disk space in isolation, and can be installed on feature restricted embedded operating systems, such as the ‘Busy Box’ Unix operating system, or any platform supporting Python version three.

TABLE I. COMPARISON OF BITCOIN, ETHEREUM, AND REDES BLOCKCHAINS [10]

Features	Blockchain System		
	Bitcoin	Ethereum	Redes
Consensus	Proof-of-Work	Proof-of-Work	Proof-of-Signature
Participation	Permissionless	Permissionless	Permissioned
Language	C++	Go, C++, Rust	Python
Currency	Bitcoin	Ether	-
Contract Support	Limited	Full	-

B. API, Block Format, and Consensus

Making use of the Python ‘Flask’ micro-framework, Redes includes its own API with 6 initial functions: register a node, remove a node, trigger consensus, issue a transaction, create a block, and request the longest chain. Testing functionality and interacting with the underlying ledger is done through calls to these 6 API’s.

Although the API format is consistent with other blockchain projects, the larger change is the format of the ledger blocks themselves. The block fields in Redes do not include provisions for currency or account balance as in

Bitcoin and Ethereum. These are replaced with ‘mac address’ and an ‘action’ field for SON control of network access in the proof of concept use (3). Table 1 shows a high-level comparison of Bitcoin, Ethereum, and the new Redes blockchain.

def new_transaction(self, sender, recipient, mac, action) (3)

As a permissioned system, the ‘register a node’ API allows initial registration of peers in the Redes network. Only known peers are allowed to participate in synchronization, with its registered node IP serving as signature in synchronization requests. This IP whitelisting gives Redes its proof-of-signature consensus name. Network synchronization is handled by either an API request to ‘request the longest chain’ for requesting the chain of a single peer, or through the ‘trigger consensus’ API, which notifies all known peers to request the chains of their known peers. Algorithm 1 shows the pseudocode each node uses to reach consensus, based on the longest valid chain received after a request to the trigger consensus API.

Algorithm 1: Pseudocode for Redes Proof-of-Signature Consensus

```

1. function resolve_conflicts(self)
   // constructor
2.   neighbors ← known nodes
3.   new_chain ← null
4.   old_chain ← local chain
5.   max_length ← block length of local chain

   // network synchronization
6.   for node ∈ neighbors
   // request chain of all neighbors
7.     response ← requests.get(https://{node}/chain)

8.     if response != null
9.       length = response.length
10.      chain = response.chain

   // if chain is longer and also valid,
   // replace the local chain
11.    if length > max_length and self.valid_chain(chain)
12.      max_length = length
13.      new_chain = chain

14. function valid_chain(self, chain)
   // constructor
15.   last_block ← chain[0]
16.   current_index ← 1

17.   while current_index < length(chain)
18.     block ← chain[current_index]

19.     if block[hash] != self.hash(last_block)
20.       return false

21.     if !self.valid_proof(last_block[proof], block[proof])
22.       return false

23.     last_block = block
24.     current_index + 1
25.     process_son(block)

```

For this research, each node issuing a transaction, was required to hash the block itself, and after, trigger consensus on the network in order to remove the compute race condition present in Bitcoin and Ethereum. Before a node accepts the transaction, the proof is still validated before the updated chain record is accepted. Integrity and consensus of state formed in the chain is still assumed valid, as any modification or corruption of previous block data changes and invalidates the hash result achieved by network nodes when the proof is checked (fig. 3) [10].

Fig. 3. Forward hash linking in Blockchain Data Structure [10]

C. Decentralized Network Access Control Use Case

To test initial function of the Redes blockchain prototype, a testbed was devised, consisting of 3 802.11 capable wireless access points running the OpenWRT operating system, and installed inside VirtualBox. Running Redes within an installation of OpenWRT, the combined system, inclusive of the operating system, web server, and database - totals less than 60Mb for the VirtualBox disk image. Other specifications for the OpenWRT hosts are 1 virtual CPU core and 256Mb of RAM. The target of the testbed was to prove an early application of Redes state consensus, combined with local application control to execute the decentralized SON function of network access control among the otherwise isolated wireless access points.

Fig. 4. New node registration using the Postman utility and Redes JSON API

Beginning with the base OpenWRT image, the three systems had all dependencies installed, then set to run the Redes blockchain. After this initial validation, the Redes blockchain API was validated using the 'Postman' API testing application. All nodes were registered with each other, using

the 'register a node' API function, to allow syncing and writing the Redes blockchain (figure 4). Each OpenWRT system then issued 'create a block' transactions to the Redes JSON API interface, with a value filled for 'mac' and an additional 'action' field, signaling a device should be 'allowed' or 'denied' network access (figure 5). In total, validation of the 6 API functions was successful, proving that the desired data was stored in the chain and consensus based on the longest chain could be formed - although network access was not yet tied to information stored in the chain.

Fig. 5. Issuing a new block using the Postman utility and Redes JSON API

A final test from the testbed required pairing local code execution with the Redes state consensus, allowing the OpenWRT systems to issue system commands to add and remove devices from its local firewall configuration - in turn permitting access to the previously isolated wireless networks (algorithm 2).

Algorithm 2: Pseudocode for SON Local Execution Within OpenWRT

1. *function process_son(block)*
 // constructor
2. *device_mac* ← *block(mac)*
3. *device_action* ← *block(action)*
4. if *device_action* = *add*
 // openwrt platform specific commands
 openwrt subprocess.call('add firewall rule [device_mac]')
 // apply configuration to the last firewall rule added
 openwrt subprocess.call('set firewall.rule[-1].target=accept')
 openwrt subprocess.call('set firewall.rule[-1].proto=tcp udp icmp')
 openwrt subprocess.call('set firewall.rule[-1].src=lan')
 openwrt subprocess.call('set firewall.rule[-1].src_mac=[device_mac]')
 openwrt subprocess.call('commit and reload')
5. if *device_action* = *remove*
 openwrt subprocess.call('delete firewall rule [device_mac]')
 openwrt subprocess.call('commit and reload')

Running the updated Redes code successfully allowed adding new blocks, forming consensus based on the longest

valid chain, and finally adding the new devices to local firewall rules, demonstrating a decentralized SON use case of federating access controls to the additional OpenWRT devices operating the Redes blockchain.

V. DISCUSSION AND FUTURE WORK

In this research, we outline popular applications of the blockchain data structure and the limitations tied to its currency and contract use. Next, we outline the need for application specific blockchain technologies, presenting our SON use case where execution of code differs among nodes, and value is derived from the blockchain data itself, rather than a cryptocurrency token. Finally, we detail and demonstrate the Redes blockchain, a new blockchain we've developed which uses a proof-of-signature consensus to reduce hash power requirements and handle a single decentralized SON function of federating access controls among otherwise isolated 802.11 networks.

Redes validates a path for use-case specific blockchain, rather than use of existing chain ecosystems for machine communications in our SON specific use case. The testbed delivers a basic permissioned blockchain that can be used to share and action data, while consuming less resources by removing the compute race condition inherent to Bitcoin and Ethereum POW; replacing it with a proof-of-signature consensus better suited to permissioned network environments. In present form, the proof-of-signature consensus does not include provisions to handle malicious or malfunctioning nodes, and as a result, it is understood to be vulnerable to forking originating from block timing and/or forging of transactions using spoofed IP addresses as signature. Future research for Redes will focus on development of its security and consensus model, including but not limited to controls for block timing and collisions, implementation of cryptographic key signatures, and support

of additional wireless SON functions, such as neighbor discovery, power control, and wireless channel selection.

REFERENCES

- [1] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 1–9. <https://doi.org/10.1007/s10838-008-9062-0>. 2016.
- [2] Andreas M. Antonopoulos. 2014. Mastering Bitcoin: Unlocking Digital Crypto-Currencies (1st ed.). O'Reilly Media, Inc..
- [3] William Feller. An introduction to probability theory and its applications. "O'Reilly Media, Inc."; 1957.
- [4] Vitalik Buterin. A next-generation smart contract and decentralized Application Platform (White Paper). *GitHub*, 223. <https://github.com/ethereum/wiki/wiki/White-Paper>. Accessed 23/05/2019.
- [5] Privatix Network; 2018. <https://privatix.io/>. Accessed 23/10/2018.
- [6] Magdalena Nohrborg. 2019. Self organizing networks. <https://www.3gpp.org/technologies/keywords-acronyms/105-son>. Accessed 23/05/2019.
- [7] Juan Ramiro, Khalid Hamied. Self organizing networks; self planning, self optimization, and self healing for GSM, UMTS, and LTE networks. "O'Reilly Media, Inc."; 2012.
- [8] Qualcomm Technologies, Inc. 2019. Qualcomm Wi-Fi SON and distributed networking. <https://www.qualcomm.com/solutions/networking/features/wi-fi-son>. Accessed 1/5/2019.
- [9] Oliver, M., & Majumder, S. (2019). Motivation for TV white space: An explorative study on Africa for achieving the rural broadband gap. In *The 2nd Europe – Middle East – North African Regional Conference of the International Telecommunications Society: "Leveraging Technologies For Growth"*, February 18-21, 2019, Aswan, Egypt.
- [10] Nutthakorn Chalaemwongwan, Werasak Kurutach. 2018. State of the art and challenges facing consensus protocols on blockchain. *International Conference on Information Networking, 2018-January*, 957–962. <https://doi.org/10.1109/ICOIN.2018.8343266>.
- [11] Steven Platt, Miquel Oliver Riera. A distributed ledger-enabled interworking model for the wireless air interface. IEEE 5th World Forum on Internet of Things. April 2019.