© 2024 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Holochain-Based Secure and Energy Efficient IoT Network

Ambono M. Kple [1], Deepak GC [1], and Bhaskar P. Rimal [2]

[1]School of Computer Science and Mathematics, Kingston University London, KT1 2EE, United Kingdom
[2]Department of Computer Science, University of Idaho, Moscow, ID 83844, USA

***Abstract*—The security and performance of a microservice based IoT platform rank among the top significant factors impacting the IoT clients within the network. It is also the case that the energy consumption of such a network has become a critical topic and is greatly under scrutiny. Therefore, it is necessary to design the network to protect against cyberattacks and keep energy consumption at the lowest possible level while maintaining higher system performance. A new Holochain architecture for IoT security is proposed in this paper. A Blockchain network memory usage and energy consumption for IoT data transmission are compared side by side with a Holochain-based IoT-microservice platform. The obtained results demonstrate that the proposed Holochain framework achieves lower energy consumption, higher performance and better scalability than the Blockchain network.**

I. Introduction

The Internet of Things (IoT) is a network of physical devices, known as things that collects data from diverse locations through the Internet and processes them to extract information. It is described as the most emerging technology that allows distributed objects to talk to each other [1]. With the rise of cloud computing, new technologies have flourished to improve device functionality and performance or add more security layers to the network. In the meantime, microservices [2] as scalable and sustainable approaches for software development are increasingly becoming the backbone of a huge web of platforms as they turn into the favourite development pattern among modern software developers with multiple design approaches [3] and new architectures [4].

A meta-framework for operations involving IoTs should primarily include innovative distributed systems, security and performance-boosting layer solutions. Blockchain has positioned itself as a first-choice platform for the architecture of IoT systems. However, it was found that as each entity of a Blockchain network stores users' transactions in a chain, the memory requirements when Blockchain entities escalate with increasingly longer chains, thereby jeopardizing practical applications of resource-constrained IoT devices [5]. Furthermore, there are other consequences when increasing transactions in a longer chain, e.g., a considerable bandwidth will be required in the near future. Together with data sharing, this gives rise to security vulnerabilities among which denial of service (DoS), replay, link spoofing and forced delay will represent the biggest challenge because of their lack of cyber threat countermeasures.

It has also been found that there is a requirement for additional computational energy for mining and consensus algorithms [6], which is described as a blockchain mining scheme with its energy consumption and waste due to the requirement of enormous computing power. Furthermore, because access and visualization to the client often mean authentication and authorization across a complex network of IoTs, credentials must be carried out across platforms. It follows that latency and low performance are also often major setbacks besides high energy consumption requirements for a Blockchain network. Therefore, this paper investigates a new Holochain network approach with regard to improving on the issues aforementioned. The approach requires coding server-side applications and doing relevant configurations necessary to run a Blockchain and a Holochain network. Ordinary client applications customise for IOTs, i.e., small electronic devices interact with each network during concurrent simulation where traffic data, energy usage and memory consumption are collected and analysed in each case.

II. Blockchain and Holochain for IoT Security

The discovery of Blockchain technology means the discovery of an ideal candidate to create a safer and more secure system for IoTs [7]. As a result, it is possible to create a blockchain-based IoT system that is more transparent and secure and should bring onto the market a generation of identical architecture. However, Blockchain technology may not be appropriate to solve the majority of IoT security issues [8]. This is because the size of data flowing in a Blockchain system overwhelms the static memory of any IoT device. Also, the Blockchain mechanism relies on a proof of work consensus algorithm, which requires an enormous amount of memory as well as huge energy consumption. Given these two conflicting appreciations of Blockchain [8], it appears necessary to establish how IoTs could be made safer with a robust technology which is superior to a one-way function hashing such as blockchain without being trapped by the blockchain downsizes.

Holochain is an emerging technology defined by the Holochain foundation [9] as a framework for building distributed peer-to-peer applications that take an agent-centric approach. Unlike in Blockchain [10] the Holochain performs the task by combining two underlying techniques, which are *(i) distributed hash table (DHT)* and *(ii) hash chain*. The DHT is focused on data propagation issues, and hash chains are built to preserve data integrity. The primary aim of Holochain is to provide more flexibility to agents in the chain; thus, the name is agent-centric. It means there will be no dominance

and restrictions from a server-side resource, which effectively provides a completely distributed network. The DHT replaces the need for centralized flow control or data management. DHT can be implemented and utilized in IoT networks for storing the chain of transition data in each node to ensure the autonomous nature of a Holochain-based network.

A large number of solutions have been proposed, such as DNS-based dynamic authentication for microservices-based IoT [11], identity-based encryption [12] and mutual authentication with multi-factor [13]. In many cases, a combination of traditional cryptographic methods is made to achieve better results, i.e., three leading cryptographic technologies make up the backbone of the Holochain framework. They are hash chains, cryptographic signing and a DHT. This paper will further evaluate how Holochain could be a possible framework to solve IoT network security challenges.

III. System Model

Holochain is a highly scalable and energy-efficient distributed ledger system. It provides a solution to scalability issues through its underlying DHT technology and does not incorporate mining or staking operations on its network. Holochain nodes do not need to transfer large amounts of redundant data. It shards data, i.e., horizontally partitioning of the database. As a result, the nodes require less bandwidth and do not overload the network where Blockchain causes a congested network, resulting in slow transactions and higher energy consumption.

1) Client Apps: The client application is at the nucleus of the entire system and provides a link to all the other components of the network to make them work together. Fig. 1 shows the IoT accessing the DHT and sending data to it. The application sets the validation rules, ensuring data is pristine and unmodified. Many programming languages are available for the application. e.g., Lisp and Typescript, amongst others.

2) Local Hashchain: The local Hashchain or source Hashchain is the key element that allows the client app to work offline. It is a local ledger that each IoT must possess. The data in the local Hashchain must be signed off before being authorized in the global shared DHT. Precisely, this is what happens when two nodes interact. They must validate their data individually by signing it in their local Hashchain before sending it to the Global DHT. Making only the nodes in transaction validate data is the fundamental idea of the Holochain revolution. It helps significant energy savings compared to the blockchain, where all the nodes in the network are invited to validate data for transactions they are not necessarily part of.

3) Distributed Hash Table (DHT): The DHT is shared storage, which plays a crucial role in the Holochain framework. The DHT allows IoTs to validate their data using cryptographic hash functions. Consequently, each client app on the IoTs is preserved with a validated signature confirming data originates from a local Hashchain. DHT maintains not only bilateral transactions but also multi-party transactions between IoTs. As such, all the parties benefit from transactions from each other, and metadata is created, improving the performance. Whenever a hash value does not match during the transaction, it is automatically rejected, and other nodes are alerted via a gossip protocol, which ensures collective security.

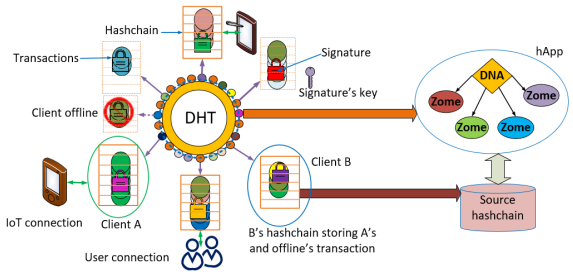

Fig. 1. The considered system model of Holochain-based IoT security.

4) Holochain Applications (hApps) or DNAs: The Holochain Applications (hApps) also called the distributed network applications (DNAs) are the server-side applications with the core logic and rules. The DNAs are usually written in a programming language called *Rust*. The DNA entry contains the functions that define the operations on the network. Those functions are called zomes and are domain-specific. They are dedicated to creating, retrieving, linking and transforming data intended for the client apps. Commonly DNA entry packages contain zomes for basic create, retrieve, update and delete (CRUD) operations as well as for the domain-specific validation rules such as for how data is stored, shared and processed. Therefore, each client app connection to the Holochain server is validated through the DNA.

In the considered system model, the Holochain framework is designed as shown in Fig. 1 for further investigation. Fig. 1 shows how transactions are carried out between a Holochain server application called hApp and the client node. The client sends requests to the hApp with a signature key alongside. This can be done both online and offline.

The Holochain approach is appropriate for the proposed system because peer validation and authentication via signature and cryptography signing ensure a high degree of security when accessing various IoT services. The scalability with a large number of users or IoTs connecting to the chain makes it attractive. Furthermore, no mining means there is no high level of energy consumption. As a result, this is a highly eligible candidate for IoT usage in a microservice environment.

IV. Proposed Method

This section describes the algorithm used to create a Holochain network that includes an hApp composed of several Zomes that multiple client apps or IoTs can call. The section presents three algorithms, i.e., the hApp Algorithm, node activities Algorithm and client-side Algorithm.

A. Holochain Framework Transaction Algorithm

Algorithm 1 shows that the Holochain framework works by initializing a local Hashchain. The user can create their Zomes or hApps once their local Hashchain is created. Subsequently, once the connectors are started, the node can connect to the

Algorithm 1 Holochain Framework Transaction Algorithm

Step1: Initiate Agent Network

n: number of nodes
Initiate local Hashchain H_l
Create private key Pr_k.
Create signature S_{app}
Create entry validation rules V_{vrn}

Step2: Create hApp Tools

Function: Create validation rules V_{vrc}.
for ($i = 1$; $i \leq n$; $i++$) **do**
    Set security feature (DNA)
    End function: Create validation rules
    Function: Verify signature
    Verify rules with hApp signature S_{app}
end for
End function: Verify signature
Function: Data journey
if (local data) **then**
    Record on the local hash chain
else
    Share on DHT
end if
Consensus validation for rules
End Function: Data journey

Algorithm 2 Node Activity

Step1: Node Activity

Validate DHT
Shard data, i.e., divide data into small chunks
Get data from DHT
Store data in DHT
Store data in peer entry
Node Security rule
Validate data against source chain publisher
Sign own action against own private key
for ($i = 1$; $i \leq n$; $i++$) **do**
    Validate against each other
    Check against neighbours
end for

Step2: Attack Immunity and Gossip

if (fake or no validation rule) **then**
    Reject the transaction
    Set as an invalid entry
    Initiate gossip protocol with neighbours
    Record cryptographically signed proof of the behaviour of the attacker
    Assess the membrane/audience
    Decide level of communication restriction
    if (the membrane is for restricted/closed group) **then**
        Stop communication with attacker agent
    end if
end if

local Hashchain, which will receive a validation message. Among received messages, some might be intended attack vectors. The node will start an immunity mechanism called *gossip* and figure out that it is being attacked, and propagate details of the attacker to neighbours.

1) Initiate Agent Network: As an open-source framework for creating secure, reliable peer-to-peer applications, Holochain connects user devices using locally installed software. This means the initial creation of security tools for the user. The user must create and initiate a local Hashchain H_l, called source Hashchain. Being agent-centric means, Holochain directs each agent or user to store their transactions in their Hashchain. This is understood as Holochain pushing to the edges rather than centralizing. Each transaction authored by an agent, such as a posted message or a response to a request, is stored in the agent Hashchain and rendered temper proof with a signature S_{app} created with the agent private key Pr_k. When the local Holochain software is installed, DNA becomes a validation rule V_{vrn} shared among other Peers. In this way, when an agent hacks into a peer's local software to insert malicious software, this will be rejected by other peers, which will compare the original DNA with the new one each time they receive a message.

2) Create hApp Tools: Once the local Holochain software is installed on a local device, the ground is laid for creating any Holochain application (hApp) type. As mentioned in the previous section, for a hApp to be safe and secure, dispositions have been taken, including the local client's validation rule (DNA), the hApp signature S_{app}, and the rule for the data journey. Each peer or node will have its own validation rule V_{vrn_i}, created at the node's registration in the Holochain network. When a hApp is created on the node, it is automatically appended to the node's validation rule (DNA). The DNA is also saved on the DHT of the Holochain framework. When an entry is made on a node, it can follow two data journeys. If the journey is a request to share with other nodes, it must be signed first, and the validation rule will be added. There is a consensus for the validation rule when messages are shared among nodes. At The DHT level, the saved DNA of the sender node is compared with the incoming DNA. If this is not tempered, the DHT will allow the incoming DNA to be shared whilst recording it. However if the journey is meant to be local only, no check is made at the DHT level, and it remains on the originating node.

3) Node Activity: At the node level, three activities are likely to occur: originate and keep data locally, share data among other nodes, or receive data. In the second case, the node validation rule V_{vrn} is applied to the data signed with the node's private key Pr_k. If the message is too big, it is sharded, i.e., divided into smaller messages. The data is then sent to the intended node entry and stored in a random node's local Hashchain to be accessible when the originating node goes offline. The message is also stored in the DHT. In the third scenario, where the nodes receive data from other nodes, the receiving node can validate against the sending node because its validation rule is attached to the message and stored in the DHT. The node can get the sender's, also called neighbour's, validation rule from the DHT and use it to validate against the message's DNA. If the validation fails, the transaction is rejected and classified as an attack. A mechanism is known as *gossip protocol* which immunises the entire network against attacks is triggered.

B. Security Module

The Algorithm 2 includes a mechanism for Holochain security. It is called the attack and immunity module. The Holochain network is a peer-to-peer network. Nodes can communicate on a peer-to-peer basis or by node detection via bootstrapping. This is where a node establishes a connection with another one in order to communicate with a second node connected to the first. On integrating the Holocahin network each node needs to commit its DNA to the centralized DHT. The node's DNA entry is hashed to produce a unique identifier known as header hash used as a key to store and retrieve the entry in the DHT. Any new entry of the node is stored in the DHT with a timestamp and the header hash. The result is a cryptographic signature of the node's entry in the DHT. Because entries are cryptographically signed and validated with the node's DNA, other nodes can trust the DHT. As a result during a peer-to-peer communication, the requesting node fetches data with the sharing node whilst querying the DHT on the validity of the data. If the shared data does not match any entry in the DHT the data is automatically flagged as a malicious entry alongside the author's info. Therefore nothing can tamper with, control, or stop communications with nodes. Online safety is ensured through mutual accountability. Each piece of data has a cryptographic audit trail associated with its author, and each user helps enforce common application rules and identify bad actors. In the case of an attack or fake validation rule, the transaction is rejected and the transaction is stamped with an invalid entry tag.

In the next steps, the strength of the attack is measured against the network's membrane. Each network has specific rules and its own access control called membrane implemented via Zomes. Similar to all access control rules, membranes also have levels. Criteria for membranes are defined based on various factors such as node reputation, identity of the author, or the hApp's requirements. In general, the more sensitive or confidential the information is on the network, the higher the membrane. When there is an attack the Holochain mechanism decides the level of communication appropriate for the attack. This depends on the specific requirements of the network. This can result in stopping all communications with the node from which the attack comes, as far as the network is concerned it is the attacker. An example of an attack is a Sybil attack [14]. It is an attack on a computer network service by creating many pseudonymous identities using a single node within a peer-to-peer network. When an attack occurs the DHT receives a message which will be examined against the audit trail of information sent to it by the node. If a suspicious behaviour is detected the DHT will not validate the entry but instead reject it. This results in creating a cryptographically signed proof of the behaviour of the attacker. It hashes information about the attacker's behaviour and the rejected data. Then start a gossip protocol with neighbor nodes which is the cryptographic signature of the attack being shared to nodes in the network. Subsequently, the network will consider its specific requirements for what it is designed for. Peers will reject the incoming agent with the malicious or invalid key when it tries to connect to the membrane. Having a membrane proof is what protects peer-to-peer applications from Sybil attacks. According to the rules of the membrane nodes without valid membrane proof can be excluded, isolated or deselected.

Algorithm 3 Client Side Application

Initialize the Client app to connect to hAPP
Set up connection variables X_i of hApp to connect to specific client, where $i = 1, 2, \ldots, n$
Input variables: k_i
Output variables: p_i
Create a function to connect to conductor
Function: Call hApp

1: Define the agent public Key PP_k
2: Connect with variable X_i and PP_k
3: Set failure handling condition
4: Define the context of the request $R_{(hApp)}$ with appropriate connection variables X_i
5: Make the request with the context of request $R_{(hApp)}$

End Function

C. Client Side Application Algorithm

Algorithm 3 describes a Holochain application deployed on a node interacting with other nodes. Nodes that communicate with other nodes use a client app that calls the other node's hApp's conductor, which is an hApp's connection point. The client app needs to know information, including the port and API name of the hApp they are interacting with and the name of the hApp's function they are trying to invoke. Therefore, the client app must set the variable X_i for the hApp's input and output parameters. The app function that handles requests to the server defines the node's public key used for signing. The app's function is a standard JavaScript function. It only needs to set the hApp input parameters and configure its expected output parameters. During the call, the Holochain validation algorithm will be triggered, and the client will receive the appropriate output when the validation is successful.

V. Simulation Results

A. System Requirements for Holochain

The Holochain framework can be deployed on different operating systems, such as Windows, Mac, and Ubuntu. However, Holochain is a peer-to-peer network, and to connect, the users need to install the Holochain software on their devices. Two possibilities are currently available. In the first scenario, *Holo* is software that can be deployed on users' machines to bridge the mainstream Internet and Holochain. In return, users can collect Holofuel, which is not exactly Bitcoin but is highly comparable. In the second scenario, *Holoport* is a hardware device to which other devices can connect to access the Holochain. This paper has implemented the traditional approach of peer-to-peer connection using downloaded software. First, the Holochain software has been installed on a 64-bit Linux (Ubuntu 20.04-12) machine with 255GB of storage and a memory of 20GB where the hApp was created.

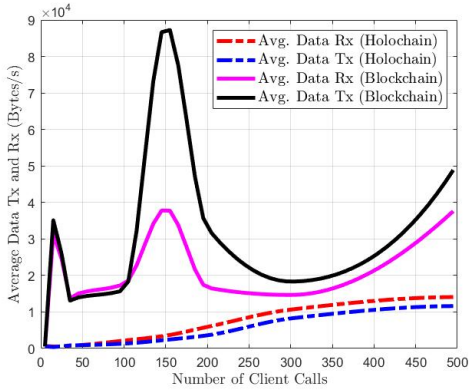

Fig. 2. Analysis of data transmitted and received between IoT devices and Holochain Apps, and blockchain nodes.

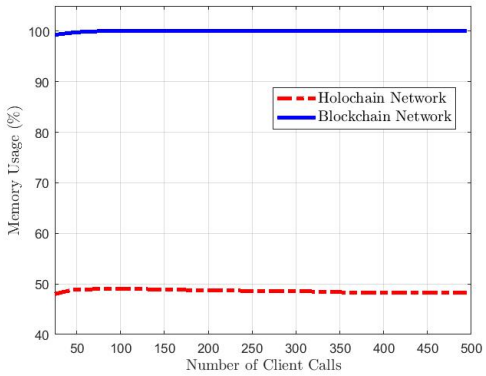

Fig. 3. Analysis of memory usage for various client calls.

B. System Requirements for Blockchain

The Blockchain framework can also be deployed on different operating systems, such as Windows, Mac and Linux. As a result, blockchain was deployed on the same operating system as Holochain, which is a 64-bit Linux (Ubuntu20.04-12) machine with 255GB of storage and a memory of 20GB.

The simulations for Blockchain and Holochain are conducted by sending multiple requests to the server. For Holochain, this is done from client apps to an hApp which is the server side. The hApp sends the response to the client apps. The experiment is conducted multiple times. Each experiment simulates 500 user calls. Data sent, data received, memory usage, and energy consumption are computed using the average value of all the simulation experiments. For the blockchain 500 calls, which are imposed by resource limitations, are sent by client nodes, and these requests are processed until the response is available and sent back to the client.

C. Network Usage

We analyze and monitor the network traffic of a Holochain infrastructure determine how it handles the network load and collect data using an application such system monitor available for an Ubuntu server. We now monitor the average bandwidth during the sending and receiving process of all the experiments conducted during the simulation.

1) Data Transmitted: In both Holochain and Blockchain cases, data sending is simulated for a client node via a software application. It is configured with a payload that is suitable for an IoT device such as a Raspberry Pi.

The transmit and receive data analysis is shown in Fig. 2 for a Proof of Work based Blockchain in which consensus is achieved by each node validating all transactions in the network. The first 200 calls follow a slow ascension. There is inflexion at 200, and between 200 and 300 calls, the slope is even steeper, and the line becomes an exponential curve. There is another inflexion at 300, after which the line tends to be a logarithmic curve. We can assume that during the first 200 calls, 8000 Bps are consumed in the network. This is a learning stage for the network where data is being studied and grouped into metadata. After 300 calls, once the metadata is structured, the bandwidth becomes more stable at around 1200 Bps. It is an important feature that makes Holochain more scalable. For the Holochain, data transmitted remains very low from the beginning, then increases slowly with the number of calls. It can be observed that data transmitted for Holochain remains far below that of Blockchain for any number of calls. Data transmitted is lower for Holochain. This is because the data needs to be verified only in the hashtable and in a couple of nodes.

2) Data Received: The data received curve for Blockchain is similar to the data transmitted curve. It follows the same pattern described for the data transmitted. Each payload is delivered with proof of work increasing the size of the data more than Holochain. For the Holochain, data received follows an identical pattern to its data transmitted curve. It starts very low from the beginning, then increases slowly with the number of calls. It can be observed that data received for Holochain remains always lower than that of Blockchain for any number of calls.

D. Memory Usage

Memory usage is the average amount of storage space set aside during data transfer between the client app and hApp, expressed in percentage. In order to investigate the memory usage in the Blockchain architecture, a simulation of 500 nodes sending each block of gradually increasing size between 10 KB and 1 MB is carried out. Fig. 3 shows that whilst for the Holochain architecture at the beginning, at around 50 calls, memory usage increases to 50% before starting to fall even when the call reaches 500. The Blockchain memory usage seems to hit 100% on the first few calls and remains there for the rest of the calls. This demonstrates that Holochain uses less memory than Blockchain.

A well-managed memory is essential as this allows for a huge number of clients to connect. Holochain framework is scalable; multiple clients can connect without creating a bottleneck, which is an excellent and fundamental general feature. Creation of reusable metadata and unnecessary request avoidance reduce memory usage and CPU cycle per bit,

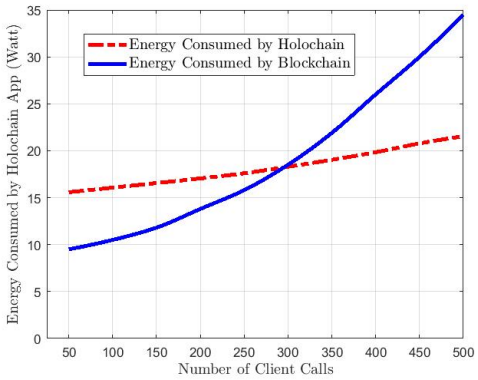

Fig. 4. Analysis of energy consumption for various client calls.

as shown in Fig. 3. The scalability is even greater with the relatively low memory usage in a Holochain network. With blockchain, for instance, as the network grows, keeping identical copies of the database causes an issue. Still, with Holochain, as each agent has its database and DHT coordinates all databases, this problem can be significantly improved.

E. Energy Consumption

We study the average energy consumption in both network architectures using an application such as Power Statistics available on a Ubuntu server. During the data transfer, it can be seen in Fig. 4 that the energy consumption for Blockchain is below that of Holochain for the first 290 calls and rises exponentially after that. The display of both curves falls in a behaviour called the deception of linear vs. exponential. Holochain with the linear curve is on top whilst blockchain with an exponential curve suddenly shows amazing growth. Given that both curves cross, it makes sense to calculate the difference in energy consumption during the 500 calls between Blockchain and Holochain to understand which one consumes more overall. This can be done by working out the geometrical area between these curves by doing mathematical integration.

We compute the difference in energy consumption by doing the computed integral from 50 to 250 $\Delta E = \int_a^b (\alpha x - \frac{x^2}{m}) dx$ and from 250 to 500 $\Delta E = \int_a^b (x - \frac{x}{\beta}) dx$. The function in each integral can be mathematically further simplified, but let's keep an eye on each separate equation. By setting α to $\frac{1}{4}$, m to 25 and β to 4 we can get mathematics models that closely match our plots. After computation, we find that the overall difference in energy consumption is around 310 W. Blockchain consumes around 310 W more energy than Holochain. It is important to notice that before 250 calls Blockchain consumes less energy. However, as the calls grow Holochain become better at energy-saving performance than Blockchain. Based on energy efficiency our results show that Holochain is more scalable than Blockchain. This is corroborated by [15] findings where they state that Holochain is in general more scalable, better performing and more efficient than Blockchain. These findings show that it might be challenging for an IoT network to support a Blockchain, whilst there could be a very good chance it supports a Holochain. Also, these results show that the Holochain ecosystem is environment-friendlier than Blockchain with lower carbon emissions. This is because nodes do not need to achieve consensus involving each node on the network as in the Blockchain case. The validation is made on a peer-to-peer basis and involves consensus only with nodes in the transaction and the DHT. Additionally, each node manages and maintains its own database. In Blockchain, an identical copy of the network data is kept by each node. This is more time-consuming and complex to execute.

VI. Conclusion

Low energy consumption for an IoT-microservice network has become a critical requirement more and more under scrutiny. Therefore, not only is it necessary to design the network to protect against cyberattacks but also to keep energy consumption low while keeping performance high. A Holochain IoT network is innovative and tackles native issues related to the Blockchain-Bitcoin and Ethereum platforms. Moreover, it implements security mechanisms by consensus between neighbours. As a result, it is energy-efficient, secure and scalable. It is the case that the energy efficiency of Holochain depends on factors such as consensus mechanisms, the number of participant nodes or network size, and the context that applies to the specific use cases. Therefore, a trade-off on performance characteristics is a reasonable approach.

References

[1] A. A. Ahmed et al. mechanism based on combiners of hash function over internet of things," *Sensors*, vol. 19, no. 17, p. 3663, 2019.
[2] E. Schäffer et al. collaborative multi-user configuration of robot-based automation solutions," *Procedia CIRP*, vol. 86, pp. 86–91, 2019.
[3] W. Jin et al. management based on independent microservices providers for gateway-centric IoT networks," *IEEE Access*, vol. 8, pp. 187 975–187 990, 2020.
[4] M. Chiang et al. opportunities," *IEEE Internet of things journal*, vol. 3, no. 6, pp. 854–864, 2016.
[5] S. Zaman et al. "Thinking out of the blocks: Holochain for distributed security in iot healthcare," *IEEE Access*, vol. 10, 2022.
[6] K. Janjua et al. architecture in fog-enabled-cloud using holochain and containerization technologies," *Electronics*, vol. 9, no. 7, 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/7/1172
[7] S. Saxena et al. secure iot: Background, integration trends and a way forward," *Journal of Network and Computer Applications*, vol. 181, p. 103050, 2021.
[8] A. Chidepatil et al. inclusive postal financial services: Blockchain and more," *Journal of Innovation Management*, vol. 8, pp. 20–27, 2020.
[9] H. Foundation. (2020) Install The Holochain Developer Toolskernel description. [Online]. Available: https://www.holochain.org/what-holochain
[10] A. C. Brock et al. distributed applications," Mar. 16 2021, uS Patent 10,951,697.
[11] D. Díaz-Sánchez et al. F. collision-based distributed and dynamic authentication for microservices in IoT," *Sensors*, vol. 19, no. 15, p. 3292, 2019.
[12] D. Pavithran et al. architecture for event-driven IoT using hierarchical identity based encryption," *Info. Processing & Mgt.*, vol. 58, no. 3, p. 102528, 2021.
[13] L. Loffi et al. "Mutual authentication with multi-factor in IoT-Fog-Cloud environment," *Journal of Network and Computer Applications*, vol. 176, p. 102932, 2021.
[14] H. Yang et al. "An overview of sybil attack detection mechanisms in VFC," pp. 117–122, 2022.
[15] S. Gaba and H. e. a. Khan, "Holochain: An agent-centric distributed hash table security in smart IoT applications," *IEEE Access*, 2023.