A Blockchain-based IoT Security Solution Using Multichain

Shereen Ismail[1], Hassan Reza[1], Hossein Kashani Zadeh[2], Fartash Vasefi[3]

[1]School of Electrical Engineering and Computer Science, University of North Dakota, Grand Forks, ND 58202, USA

[2]Department of Mechanical Engineering, University of North Dakota, Grand Forks, ND 58202, USA

[3]SafetySpect Inc., 10100 Santa Monica Blvd., Suite 300, Los Angeles, CA, 90067, USA

shereen.ismail@ndus.edu, hassan.reza@ndus.edu, hossein.kashanizadeh@ndus.edu, fvasefi@safetyspect.com

2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC) | 979-8-3503-3286-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/CCWC57344.2023.10099128

Abstract— **Blockchain technology is increasingly being utilized in Internet of Things (IoT) systems to provide secure data communications and hence prevent data tampering. Private blockchain offers improved speed and efficiency for small networks controlled by a single authoritative node while maintaining the secure, transparent, and immutable characteristics of blockchain technology. Multichain, an open-source private blockchain platform, offers high flexibility in building robust IoT security solutions through customizable configurations and permissions. Key features of Multichain include rapid deployment, unlimited asset management, and streamlined data streams. In this paper, we undertake a comprehensive review of existing work on blockchain IoT solutions using Multichain. To further validate our findings, we conduct a blockchain-based simulation experiment using Multichain and aim to provide insights into the potential applicability of the Multichain platform for securing IoT systems.**

Keywords— IoT, Blockchain, Multichain, Security.

I. INTRODUCTION

Recent Internet of Things (IoT) applications, including smart grids, intelligent transportation, smart cities, healthcare, finance, energy, agriculture, logistics, and supply chain [1], are vulnerable to unauthorized access attempts aimed at obtaining sensitive data [2], [3]. To address these security concerns, blockchain, also known as Distributed Ledger Technology (DLT), is being employed as an alternative to conventional centralized databases. Blockchain refers to a digital ledger that is distributed, decentralized, and shared among multiple parties. The use of blockchain in IoT applications has been shown to enhance security by protecting data content, ensuring secure routing, controlling access to resources, managing services accessibility, establishing identity management, fostering peer trust, and authenticating nodes [4], [5].

Blockchain is a highly desirable security solution for stakeholders from different enterprises looking to establish secure communications in a zero-trust environment. Unlike traditional centralized database systems, blockchain is designed to be resilient to failure, thanks to their decentralized ledger deployment. With every participating node having a copy of the blockchain ledger, it is often not possible to modify the shared ledger without the consensus of a certain number of nodes. By employing blockchain to secure an IoT system, trust can be established, device authentication can be performed, and data can be decentralized. The immutability of blockchain, which protects the data against malicious changes, can also facilitate quick, smooth, and cost-effective auditing and tracking of data sources [6]. Integrating blockchain with IoT motivates researchers to propose secure solutions using different blockchain platforms. Each platform has its strengths and weaknesses; therefore, adoption depends on the application's security, privacy, performance, and scalability requirements.

Blockchain can be divided into two categories: public and private. The main difference between these two categories is who is allowed to participate in the network, operate the consensus mechanism, and maintain the shared ledger. Initially, blockchain was created to provide a permissionless public network in untrusted environments, where anyone could join the network and participate in mining. However, the evolution of blockchain has shifted towards the use of permissioned private blockchains, which are more suitable for commercial and enterprise applications [7]. Private blockchains, where there is some level of trust among participants, can reach network consensus without requiring intensive computations. This allows for higher privacy, which is desirable for certain IoT applications, such as supply chain. Some commonly used private blockchain platforms for enterprise solutions include Hyperledger Fabric, Corda, Quorum, and Multichain. This study focuses on the Multichain open-source platform. The main contributions of this paper are literature review for private blockchain-based IoT systems developed using Multichain and a private blockchain network deployment and testing using Multichain which will facilitate the construction of various IoT use cases in future work, such as supply chain traceability, connecting various IoT devices, and Machine-to-Machine communication.

The organization of this paper is as follows: Section II provides an overview of Multichain as a private blockchain platform, highlighting its key features. The validation mechanism of Multichain, which utilizes the Round Robin approach, is explained in Section III, setting it apart from other platforms. Section IV showcases the utilization of blockchain-based solutions using Multichain to secure various IoT systems. The simulation experiment utilizing Multichain as a blockchain platform is discussed in Section V. Finally, the paper concludes in Section VI with future work prospects and conclusions.

II.MULTICHAIN

Coin Sciences has introduced Multichain as an open-source private blockchain platform. This platform is based on the original Bitcoin blockchain, but with enhanced functionality, enabling the management of portfolios, assets, permissions, and transactions. In contrast to Bitcoin, Multichain simplifies the configuration of several parameters, such as access permissions, chain privacy, and maximum block size [7]. Multichain is designed to be easily deployable; it is available as a portable package that can be installed by an administrator on any common operating system, including Windows, Linux, and Mac. This eliminates the need for specialized developers and makes the deployment process more accessible. Multichain version 2.0 is an updated version of Multichain 1.0 that introduces new features, such as transaction filters. Multichain, which is supported by comprehensive documentation, works with various coding languages, such as Python, C#, PHP, Ruby, or JavaScript. Multichain can be configured easily and work simultaneously with different blockchains in the same network.

Fig.1 illustrates the general architecture of the Multichain blockchain network controlled by one Admin, which sets the roles of other participating nodes. The Admin can create more Admins to avoid a single point of failure. Multichain manages access to the data using a set of participants, which obtain addresses and various permissions, allowing them to exist in the closed blockchain network. Only pre-registered participants have access to read and write blocks on the ledger. Transaction validation and block mining are performed by a known set of miner nodes, called validators. Multichain command-line interface (CLI) and JSON-RPC clients, or regular nodes, interact with the network and access the blockchain API.

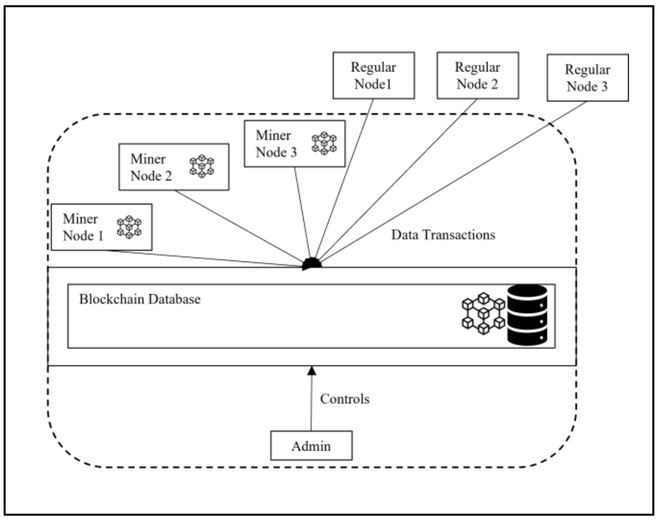


Fig. 1 General Architecture of a Multichain blockchain network

Multichain uses data streams that act as an independent append-only collection of items, which enforces shared data confidentiality. This technology is characterized by its flexibility, allowing permissions changes and delegations. Multichain is also based on round-robin scheduling, where nodes take turns creating and validating blocks and adding them to the blockchain.

III. ROUND ROBIN VALIDATION

In Multichain, block validation is achieved through a round-robin scheduling where validators take turns publishing blocks. This approach eliminates the mining race and promotes fairness within the blockchain network by avoiding conflicts between nodes competing to send transactions. Additionally, it reduces the risk of a 50% attack, as no single participant has a monopoly over the majority of block creation. The absence of cryptographic puzzles during block validation also conserves the resources of the nodes. Multichain employs the principle of mining diversity, which means that a specified percentage of miners can validate consecutive blocks without repetition. The time a miner has to wait for its turn to validate the next block can be calculated using Equation (1).

$$Interval = \# \, of \, miners * mining \, diversity \quad (1)$$

Where $0.0 \leq mining\ diversity \leq 1.0$ [8]. A value of 1 means that every miner will be included in the rotation, while 0 indicates no restrictions. Although a high mining diversity value (< 0.75) is recommended, a value too close to 1 can cause the blockchain to freeze if some miners become inactive or malfunction. To avoid this, a time limit is set to control the length of the wait, ensuring that block publication is not interrupted by unavailable nodes [9].

Round-robin scheduling aligns well with the distributed systems architecture that requires node trust, but it is not suitable for permissionless blockchain networks commonly used by cryptocurrencies. In these networks, malicious nodes can add new nodes, increasing the risk of new blocks being deployed.

IV.RELATED WORK

Several studies have leveraged the Multichain platform to propose blockchain-based IoT systems. This platform integrates seamlessly with other blockchain structures and existing systems and is compatible with common operating systems. Table I summarizes the research works that have used the Multichain platform to implement blockchain-based solutions for different IoT applications.

The authors of [10], [11], and [12] examined Multichain blockchain integrated with fog computing architecture for potential applications in healthcare, agriculture, logistics, and smart city. They conducted real experiments with smart sensors for testing and performance evaluation.

The authors of [13] conducted an experimental study to examine the use of the Multichain blockchain for a hotel booking service, testing several scenarios, and comparing latency and memory usage that might affect the user experience.

TABLE I. MULTICHAIN BLOCKCHAIN-BASED IOT SECURITY SOLUTIONS

Ref.	Year	Application	Technology	Blockchain Environment	Insights
[10]	2017	IoT	Edison Arduino boards	Linux Debian 8.5	Fog decentralized architecture integrated blockchain with smart things
[13]	2017	Hotel booking service for the tourism industry	N/A	UNIX and Windows	Three test scenarios with different numbers of nodes for performance evaluation
[11]	2017	IoT	Raspberry Pi and Edison Arduino board	Linux Ubuntu	Fog computing, IBM Bluemix Blockchain as a Service
[14]	2017	Wine supply chain	N/A	N/A	Implementation of five server nodes as a blockchain network
[12]	2019	IoT	Test bed for Raspberry Pi	Linux and Docker	Secure sensor data sharing using fog computing and blockchain
[15]	2020	Agrifood supply chain	N/A	Ubuntu 18.04 over VirtualBox	Implementation of two server nodes as a blockchain network
[16]	2020	Agro farm	N/A	N/A	Testing node creation and communication over blockchain for Agriculture farms
[17]	2020	Subscription-based data-sharing model	N/A	Windows	Blockchain with Data as a Service so users subscribe to Data Providers with a subscription plan
[18]	2020	Smart home	N/A	N/A	Proposing a smart home system of IoT devices integrated with blockchain
[19]	2020	Smart grid	N/A	N/A	Proposing an energy-based network in smart grid architecture
[20]	2021	Drug supply chain	N/A	Ubuntu	Business process testing simulation starting from the pharmaceutical industry to the end consumer
[21]	2021	Road traffic management	Raspberry Pi, Pi camera, IR sensors, RFID devices	N/A	Architectural framework of IoT integrated with fog computing
[22]	2022	The Internet of Medical Things	N/A	Windows	Hospital application architecture implementation for securing patients' clinical records

The supply chain is a critical area where blockchain and IoT can work effectively to address security issues. The authors of [14], [15], and [20] proposed the use of Multichain to implement effective blockchain-based traceability solutions for various products. They implemented several entities representing supply chain stakeholders to test scalable systems.

The authors of [17] proposed a subscription-based data-sharing model and evaluated its performance using Multichain. The authors established that Multichain ensures authorization and access control in a multi-user network.

The authors of [18] proposed a secure smart home system consists of smart refrigerator, smart home lights, IoT door security camera integrated with multichain.

The authors of [19] proposed an energy-based blockchain network for secure data sharing in smart grid architecture consists of three types of network architecture, including, home area network (HAN), building area network (BAN), and neighborhood area network (NAN).

The authors of [21] proposed a blockchain-based framework for road traffic management, where IoT devices and drones are connected to the Multichain network and with the use of fog computing.

The authors of [22] authors introduced a blockchain-based solution for the Internet of Medical Things (IoMT) to enhance the security of communication between medical devices and secure medical data within a hospital environment.

V. Blockchain-based Simulation Experiment Using Multichain

Multichain is a suitable platform for IoT applications due to its low resource consumption and storage requirements in comparison to other platforms. In our simulation experiment, we utilized Multichain and set up a single node, known as the Admin, to control the network. We provide a comprehensive explanation of the implementation and deployment steps. To demonstrate the functionality of Multichain, we established two virtual nodes that joined a private blockchain network. The nodes were able to generate transactions, read and write blocks, exchange assets, and monitor their status. The implemented architecture of Multichain is depicted in Fig. 2. Each node is assigned a node address, which is in the form of an IP address and port number, and a wallet address, which is a unique alphanumeric identifier generated when the node first connects to the chain.

A. Set-up two virtual machines

This experiment was conducted on a computer with 16.0 GB RAM and an 11th Gen Intel(R) Core (TM) i7-1165G7 @ 2.80GHz processor. We utilized VMware Workstation 16.0 to run two virtual machines (VMs) with Ubuntu ISO image 12.04 64-bit architecture and 4.0 GB of RAM each.

After setting up the two Ubuntu VMs, we downloaded and installed the latest version of the Multichain Community. The Windows version of Multichain Community is easily accessible at [23] and can be directly installed on a Windows operating system without the need for a virtual machine in between.

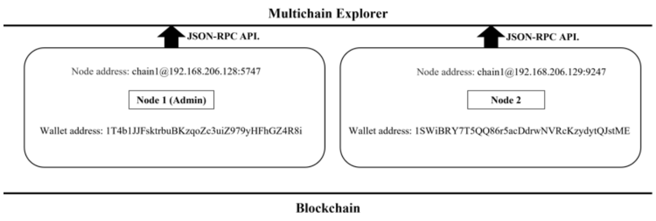
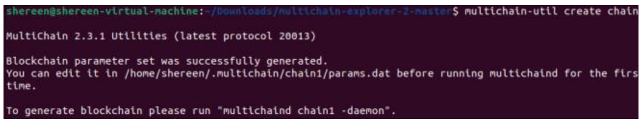

Multichain Explorer
JSON-RPC API.
JSON-RPC API.
Node address: chain1@192.168.206.128:5747
Node 1 (Admin)
Wallet address: 1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i
Node address: chain1@192.168.206.129:9247
Node 2
Wallet address: 1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME
Blockchain

Fig. 2. Illustration of the implemented Multichain architecture

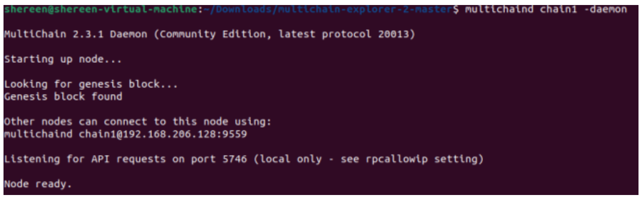
B. Set-up a blockchain

We installed Multichain Community on two VMs using Terminal CLI following the installation guide on the Multichain website, then created the blockchain network called chain1 on the Admin as follows:



```
shereen@shereen-virtual-machine:~/Downloads/multichain-explorer-2-master$ multichain-util create chain

MultiChain 2.3.1 Utilities (latest protocol 20013)

Blockchain parameter set was successfully generated.
You can edit it in /home/shereen/.multichain/chain1/params.dat before running multichaind for the firs
time.

To generate blockchain please run "multichaind chain1 -daemon".
```

By default, this node act as an Admin that could further grant other nodes Admin status. Chain1 was initialized, and a genesis block was created, as follows:



```
shereen@shereen-virtual-machine:~/Downloads/multichain-explorer-2-master$ multichaind chain1 -daemon

MultiChain 2.3.1 Daemon (Community Edition, latest protocol 20013)

Starting up node...

Looking for genesis block...
Genesis block found

Other nodes can connect to this node using:
multichaind chain1@192.168.206.128:9559

Listening for API requests on port 5746 (local only - see rpcallowip setting)

Node ready.
```

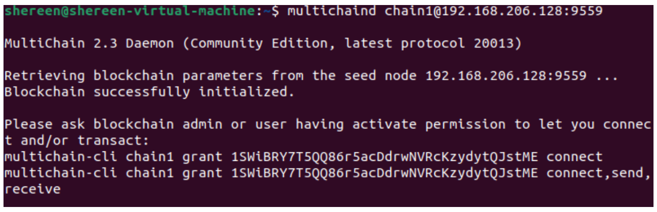
C. Multichain configuration file

All configuration parameters were set in one configuration file, as described above. This file was called "params.dat" in Multichain, created with the execution of the command multichain-util, where the Admin can control a number of nodes, user permissions, block size, and network configuration. The permissions for other nodes, such as connect, send, receive, issue, create, mine, admin, and activate, are initialized as false and will be set by the Admin; however, this permission can be set as true for all nodes while setting chain parameters, which is not recommended.

All configured and default blockchain parameters, such as chain-description, chain name, chain-params-hash, chain-protocol, default-network-port, default-rpc-port, and maximum-block-size in addition to different fields for a genesis block, such as genesis-hash, genesis-nbits, genesis-nonce, genesis-pubkey, genesis-pubkey-hash, and genesis-timestamp can be found in "params.dat".

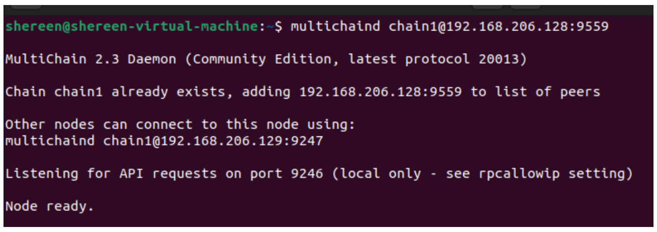
D. Connecting to a blockchain

For Node 2, we executed the following command to connect to chain 1:



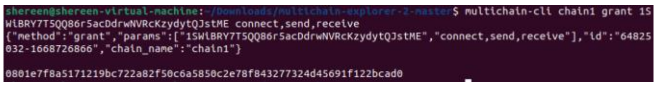
```
shereen@shereen-virtual-machine:~$ multichaind chain1@192.168.206.128:9559

MultiChain 2.3 Daemon (Community Edition, latest protocol 20013)

Retrieving blockchain parameters from the seed node 192.168.206.128:9559 ...
Blockchain successfully initialized.

Please ask blockchain admin or user having activate permission to let you connec
t and/or transact:
multichain-cli chain1 grant 1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME connect
multichain-cli chain1 grant 1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME connect,send,
receive
```

If chain 1 at node 2 already exists, the following will be returned:



```
shereen@shereen-virtual-machine:~$ multichaind chain1@192.168.206.128:9559

MultiChain 2.3 Daemon (Community Edition, latest protocol 20013)

Chain chain1 already exists, adding 192.168.206.128:9559 to list of peers

Other nodes can connect to this node using:
multichaind chain1@192.168.206.129:9247

Listening for API requests on port 9246 (local only - see rpcallowip setting)

Node ready.
```

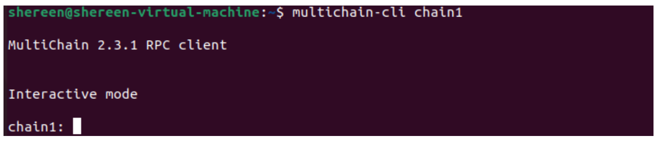
At the Admin node, the connection for node 2 should be granted as follows:



```
shereen@shereen-virtual-machine:~/Downloads/multichain-explorer-2-master$ multichain-cli chain1 grant 1S
WiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME connect,send,receive
{"method":"grant","params":["1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME","connect,send,receive"],"id":"64825
032-1668726866","chain_name":"chain1"}

0801e7f8a5171219bc722a82f50c6a5850c2e78f843277324d45691f122bcad0
```

E. Multichain interactive mode

Interactive mode can now be activated on both nodes as follows:

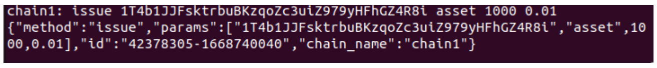


```
shereen@shereen-virtual-machine:~$ multichain-cli chain1

MultiChain 2.3.1 RPC client


Interactive mode

chain1: 
```

multichain-cli talks to the node to perform the required functions.

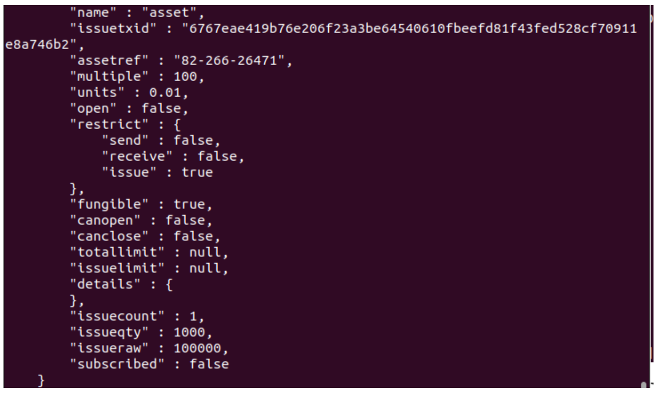
F. Issuing and exchanging assets

At the Admin node, asset1 was created with 1,000 units, each of which can be subdivided into parts using the command:



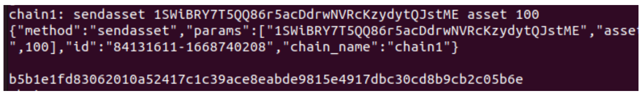
```
chain1: issue 1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i asset 1000 0.01
{"method":"issue","params":["1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i","asset",10
00,0.01],"id":"42378305-1668740040","chain_name":"chain1"}
```

At this stage, *listassets* will show the following:



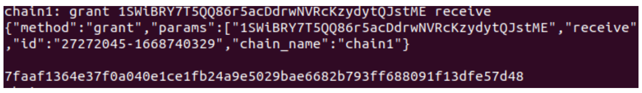
```
        "name" : "asset",
        "issuetxid" : "6767eae419b76e206f23a3be64540610fbeefd81f43fed528cf70911
e8a746b2",
        "assetref" : "82-266-26471",
        "multiple" : 100,
        "units" : 0.01,
        "open" : false,
        "restrict" : {
            "send" : false,
            "receive" : false,
            "issue" : true
        },
        "fungible" : true,
        "canopen" : false,
        "canclose" : false,
        "totallimit" : null,
        "issuelimit" : null,
        "details" : {
        },
        "issuecount" : 1,
        "issueqty" : 1000,
        "issueraw" : 100000,
        "subscribed" : false
    }
```

Then, the *sendasset* command can be used to send 100 units from node 1 to node 2. Node 1 must send the asset as follows:



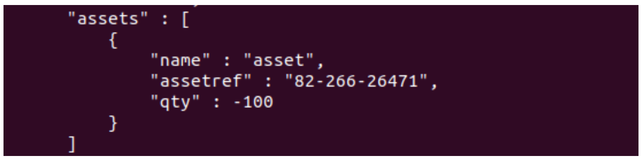
```
chain1: sendasset 1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME asset 100
{"method":"sendasset","params":["1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME","asse
",100],"id":"84131611-1668740208","chain_name":"chain1"}

b5b1e1fd83062010a52417c1c39ace8eabde9815e4917dbc30cd8b9cb2c05b6e
```

Then, receive permissions are granted at node 1 as follows:



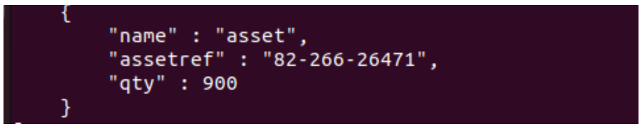
```
chain1: grant 1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME receive
{"method":"grant","params":["1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME","receive"
,"id":"27272045-1668740329","chain_name":"chain1"}

7faaf1364e37f0a040e1ce1fb24a9e5029bae6682b793ff688091f13dfe57d48
```

At node 1, we can check the asset exchange using the *listwallettransactions* command as follows:



```
        "assets" : [
            {
                "name" : "asset",
                "assetref" : "82-266-26471",
                "qty" : -100
            }
        ]
```

To check the new balance at node 1 after sending assets, use *gettotalbalances* command to get the following:

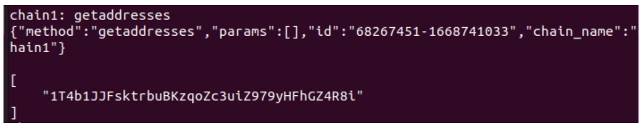


```
    {
        "name" : "asset",
        "assetref" : "82-266-26471",
        "qty" : 900
    }
```

There is no way to delete an asset once it is created.

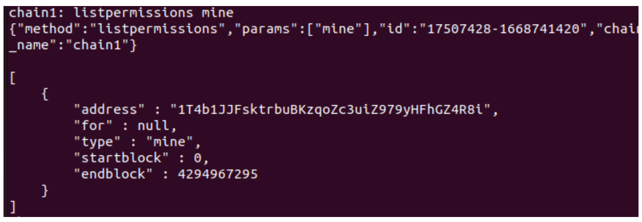
G. *Addresses in Multichain*

At the Admin node, the addresses can be checked as follows:



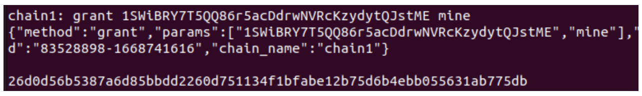
```
chain1: getaddresses
{"method":"getaddresses","params":[],"id":"68267451-1668741033","chain_name":"
hain1"}

[
    "1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i"
]
```

To check the nodes able to mine:



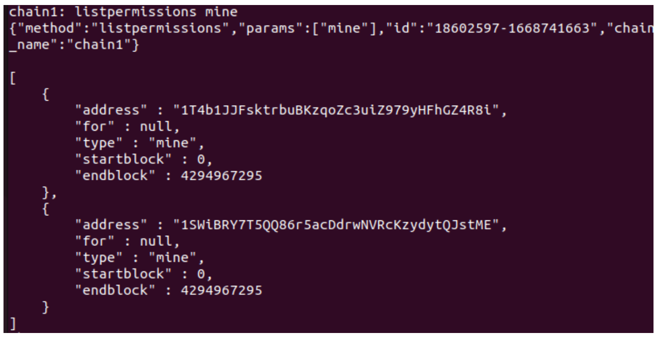
```
chain1: listpermissions mine
{"method":"listpermissions","params":["mine"],"id":"17507428-1668741420","chai
_name":"chain1"}

[
    {
        "address" : "1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i",
        "for" : null,
        "type" : "mine",
        "startblock" : 0,
        "endblock" : 4294967295
    }
]
```

To grant mine permissions to Node 2:



```
chain1: grant 1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME mine
{"method":"grant","params":["1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME","mine"],"
d":"83528898-1668741616","chain_name":"chain1"}

26d0d56b5387a6d85bbdd2260d751134f1bfabe12b75d6b4ebb055631ab775db
```

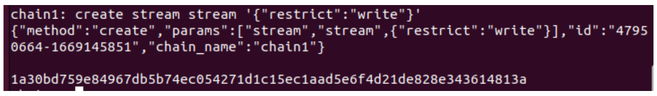
Then if we check the nodes able to mine again, we get the following:



```
chain1: listpermissions mine
{"method":"listpermissions","params":["mine"],"id":"18602597-1668741663","chain
_name":"chain1"}

[
    {
        "address" : "1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i",
        "for" : null,
        "type" : "mine",
        "startblock" : 0,
        "endblock" : 4294967295
    },
    {
        "address" : "1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME",
        "for" : null,
        "type" : "mine",
        "startblock" : 0,
        "endblock" : 4294967295
    }
]
```

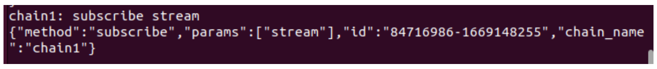
H. *Creation and subscription to streams*

Stream is an isolated append-only collection of items that allows a blockchain to be used as an immutable time-stamped general-purpose database. To create a stream at node 1:



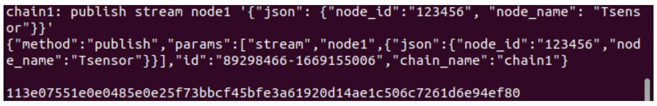
```
chain1: create stream stream '{"restrict":"write"}'
{"method":"create","params":["stream","stream",{"restrict":"write"}],"id":"4795
0664-1669145851","chain_name":"chain1"}

1a30bd759e84967db5b74ec054271d1c15ec1aad5e6f4d21de828e343614813a
```

Node 2 can be granted permission to subscribe and publish to the data stream as follows:



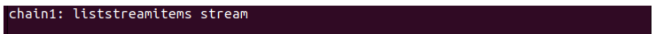
```
chain1: subscribe stream
{"method":"subscribe","params":["stream"],"id":"84716986-1669148255","chain_name
":"chain1"}
```

To publish an item to a stream, the publisher must provide a key in the form of a string, such as “node1”, and the hexadecimal data fields as an input as follows:

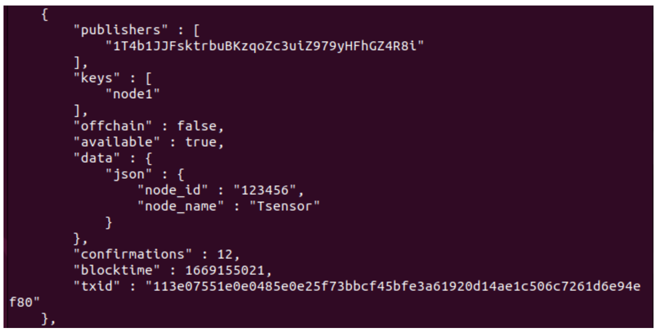


```
chain1: publish stream node1 '{"json": {"node_id":"123456", "node_name": "Tsens
or"}}'
{"method":"publish","params":["stream","node1",{"json":{"node_id":"123456","nod
e_name":"Tsensor"}}],"id":"89298466-1669155006","chain_name":"chain1"}

113e07551e0e0485e0e25f73bbcf45bfe3a61920d14ae1c506c7261d6e94ef80
```

To list the content of the stream, use the following command:

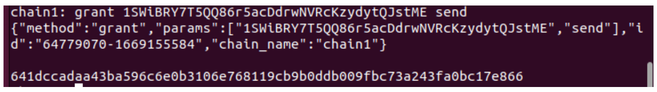


```
chain1: liststreamitems stream
```



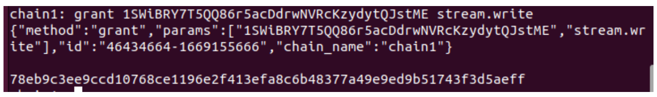
```
    {
        "publishers" : [
            "1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i"
        ],
        "keys" : [
            "node1"
        ],
        "offchain" : false,
        "available" : true,
        "data" : {
            "json" : {
                "node_id" : "123456",
                "node_name" : "Tsensor"
            }
        },
        "confirmations" : 12,
        "blocktime" : 1669155021,
        "txid" : "113e07551e0e0485e0e25f73bbcf45bfe3a61920d14ae1c506c7261d6e94e
f80"
    },
```

At node 1, node 2 can be allowed to publish to the stream using the following:



```
chain1: grant 1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME send
{"method":"grant","params":["1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME","send"],"i
d":"64779070-1669155584","chain_name":"chain1"}

641dccadaa43ba596c6e0b3106e768119cb9b0ddb009fbc73a243fa0bc17e866
```

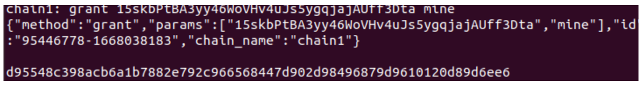
And grant stream writes as follows:



```
chain1: grant 1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME stream.write
{"method":"grant","params":["1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME","stream.wr
ite"],"id":"46434664-1669155666","chain_name":"chain1"}

78eb9c3ee9ccd10768ce1196e2f413efa8c6b48377a49e9ed9b51743f3d5aeff
```

I. Adding Miners

At the Admin node, node 2 can be given the privilege to mine as follows:

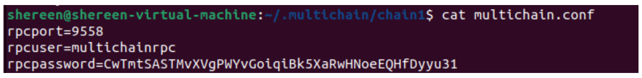


```
chain1: grant 15skbPtBA3yy46WoVHv4uJs5ygqjajAUff3Dta mine
{"method":"grant","params":["15skbPtBA3yy46WoVHv4uJs5ygqjajAUff3Dta","mine"],"id"
:"95446778-1668038183","chain_name":"chain1"}

d95548c398acb6a1b7882e792c966568447d902d98496879d9610120d89d6ee6
```

J. Running MultiChain Explorer

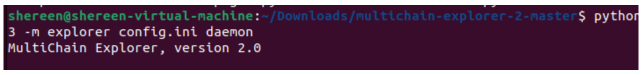
Multichain explorer 2 can be installed using the command >*wget* [*https://github.com/MultiChain/multichain-explorer-2.git*](https://github.com/MultiChain/multichain-explorer-2.git)

For the explorer to get the connection with chain 1, *rpcport*, *rpcuser*, and *rpcpassword* should be similar at both "Multichain.conf" resides at Multichain folder and "config.ini" file in the explorer folder as follows:



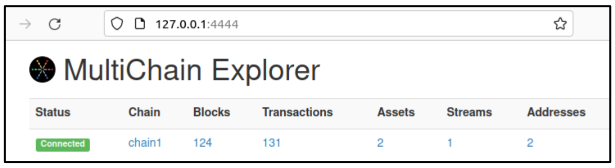
```
shereen@shereen-virtual-machine:~/.multichain/chain1$ cat multichain.conf
rpcport=9558
rpcuser=multichainrpc
rpcpassword=CwTmtSASTMvXVgPWYvGoiqiBk5XaRwHNoeEQHfDyyu31
```

Now, it is time to run the explorer as follows:



```
shereen@shereen-virtual-machine:~/Downloads/multichain-explorer-2-master$ python
3 -m explorer config.ini daemon
MultiChain Explorer, version 2.0
```

At the browser, we entered the address http://127.0.0.1:4444/ for a localhost connection as follows:

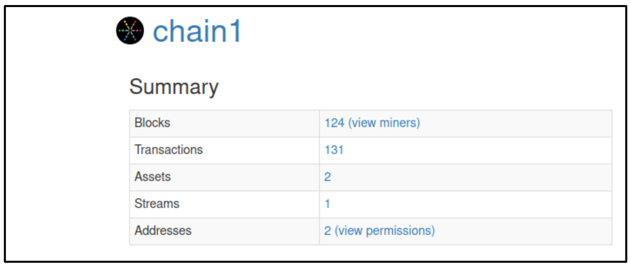


127.0.0.1:4444

MultiChain Explorer

Status	Chain	Blocks	Transactions	Assets	Streams	Addresses
Connected	chain1	124	131	2	1	2

We can explore chain1 details, which include a summary of blocks, transactions, assets, streams, and addresses as follows:

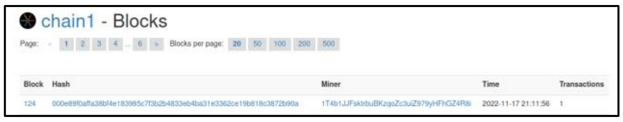


chain1

Summary

Blocks	124 (view miners)
Transactions	131
Assets	2
Streams	1
Addresses	2 (view permissions)

In addition, general information and blockchain configuration are displayed. All blocks can be checked over the network as follows:

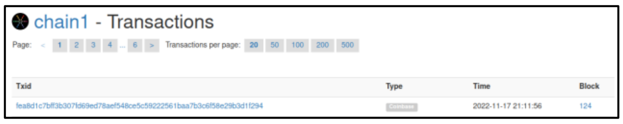


chain1 - Blocks

Page: < 1 2 3 4 ... 6 > Blocks per page: 20 50 100 200 500

Block	Hash	Miner	Time	Transactions
124	000e89f0affa38bf4e183985c7f3b2b4833eb4ba31e3362ce19b818c3872b90a	1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i	2022-11-17 21:11:56	1

All transactions as follows:

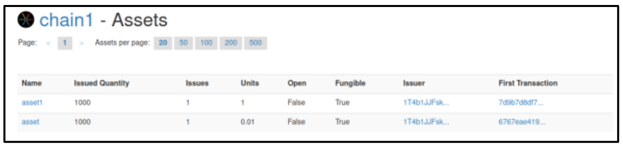


chain1 - Transactions

Page: < 1 2 3 4 ... 6 > Transactions per page: 20 50 100 200 500

Txid	Type	Time	Block
fea8d1c7bff3b307fd69ed78aef548ce5c59222561baa7b3c6f58e29b3d1f294	Coinbase	2022-11-17 21:11:56	124

All assets as follows:

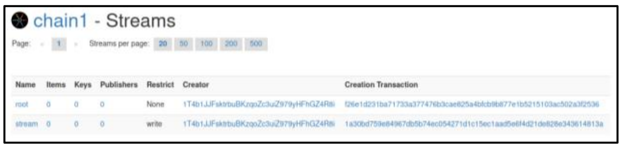


chain1 - Assets

Page: < 1 > Assets per page: 20 50 100 200 500

Name	Issued Quantity	Issues	Units	Open	Fungible	Issuer	First Transaction
asset1	1000	1	1	False	True	1T4b1JJFsk...	7d9b7d8df7...
asset	1000	1	0.01	False	True	1T4b1JJFsk...	6767eae419...

And all streams as follows:

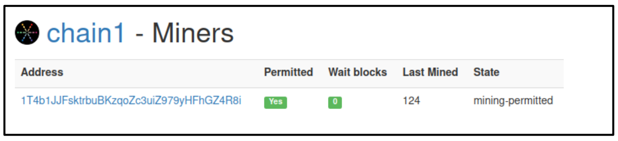


chain1 - Streams

Page: < 1 > Streams per page: 20 50 100 200 500

Name	Items	Keys	Publishers	Restrict	Creator	Creation Transaction
root	0	0	0	None	1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i	f26e1d231ba71733a377476b3cae825a4bfcb9b877e1b5215103ac502a3f2536
stream	0	0	0	write	1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i	1a30bd759e84967db5b74ec054271d1c15ec1aad5e6f4d21de826e343614813a

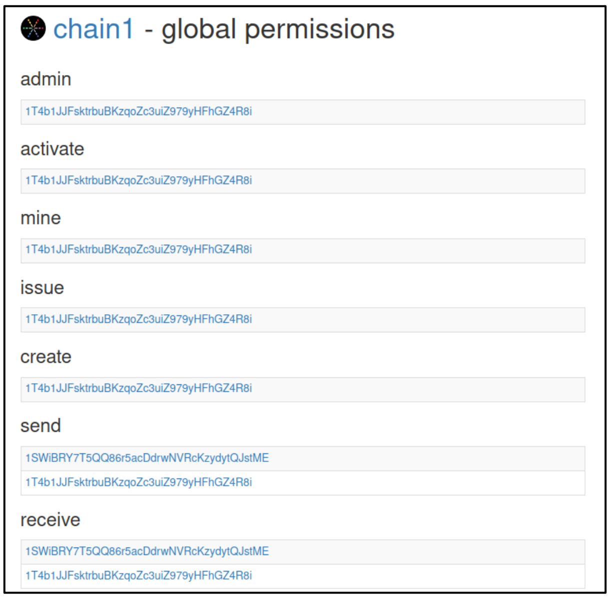
To see the details of the miner nodes, click the miners' tab. We had one miner, which was the Admin node as follows:



chain1 - Miners

Address	Permitted	Wait blocks	Last Mined	State
1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i	Yes	0	124	mining-permitted

To check the node's permissions through the explorer, we clicked view permissions to obtain the following:



chain1 - global permissions

admin

1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i

activate

1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i

mine

1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i

issue

1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i

create

1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i

send

1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME

1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i

receive

1SWiBRY7T5QQ86r5acDdrwNVRcKzydytQJstME

1T4b1JJFsktrbuBKzqoZc3uiZ979yHFhGZ4R8i

CONCLUSION

Blockchain can be categorized as public and private, and the appropriate platform was selected based on the level of trust between the participants. Multichain is designed to create and deploy private blockchains with some level of trust between a set of nodes. Multichain has several features that make it desirable for different IoT applications.

We reviewed the private blockchain IoT systems proposed in the literature using Multichain. We also built and tested a private blockchain-based network using Multichain, enabling us to construct numerous IoT use cases in future work. We aim to expand the current Multichain implementation towards an integrated blockchain-based supply chain framework. Supply chain systems, among other system-wide requirements, demands high availability, security, and scalability. Existing methods used to trace and track products in supply chain, for most part, depended on centralized data sharing systems and Radio Frequency Identification (RFID) respectively. These technologies are subject to the single point of failure, and data tampering and swapping the tags. To track and trace products in supply chain, we are planning to integrate our Multichain based system with machine learning-based scanning devices to overcoming security, availability, and scalability issues.

Acknowledgment

This work was partially funded by NOAA SBIR grant NA21OAR0210305.

References

[1] A. J. M. Milne, A. Beckmann, and P. Kumar, "Cyber-Physical Trust Systems Driven by Blockchain," *IEEE Access*, vol. 8, pp. 66423–66437, 2020, doi: 10.1109/ACCESS.2020.2984675.

[2] M. Mamdouh, A. I. Awad, A. A. M. Khalaf, and H. F. A. Hamed, "Authentication and Identity Management of IoHT Devices: Achievements, Challenges, and Future Directions," *Comput. Secur.*, vol. 111, p. 102491, 2021, doi: 10.1016/j.cose.2021.102491.

[3] S. Ismail and H. Reza, "Security Challenges of Blockchain-Based Supply Chain Systems," in *2022 IEEE 13th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2022, pp. 1–6, doi: 10.1109/UEMCON54665.2022.9965682.

[4] S. Ismail, D. Dawoud, and H. Reza, "Towards A Lightweight Identity Management and Secure Authentication for IoT Using Blockchain," *2022 IEEE World AI IoT Congr. AIIoT 2022*, pp. 77–83, 2022, doi: 10.1109/AIIOT54504.2022.9817349.

[5] A. A. Khalil, J. Franco, I. Parvez, S. Uluagac, H. Shahriar, and M. A. Rahman, "A Literature Review on Blockchain-enabled Security and Operation of Cyber-Physical Systems," in *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*, 2022, pp. 1774–1779, doi: 10.1109/COMPSAC54236.2022.00282.

[6] J. C. Song, M. A. Demir, J. J. Prevost, and P. Rad, "Blockchain Design for Trusted Decentralized IoT Networks," in *2018 13th Annual Conference on System of Systems Engineering (SoSE)*, 2018, pp. 169–174, doi: 10.1109/SYSOSE.2018.8428720.

[7] J. Polge, J. Robert, and Y. Le Traon, "Permissioned blockchain frameworks in the industry: A comparison," *ICT Express*, vol. 7, no. 2, pp. 229–233, Jun. 2021, doi: 10.1016/J.ICTE.2020.09.002.

[8] H. Tewari, A. Hughes, S. Weber, and T. Barry, "X509Cloud - Framework for a ubiquitous PKI," *Proc. - IEEE Mil. Commun. Conf. MILCOM*, vol. 2017-October, pp. 225–230, Dec. 2017, doi: 10.1109/MILCOM.2017.8170796.

[9] G. Greenspan, "MultiChain Private Blockchain-White Paper," Accessed: Nov. 17, 2022. [Online]. Available: http://coinsecrets.org/.

[10] M. Samaniego and R. Deters, "Internet of Smart Things - IoST: Using Blockchain and CLIPS to Make Things Autonomous," in *2017 IEEE International Conference on Cognitive Computing (ICCC)*, 2017, pp. 9–16, doi: 10.1109/IEEE.ICCC.2017.9.

[11] M. Samaniego and R. Deters, "Virtual Resources & Blockchain for Configuration Management in IoT," *J. Ubiquitous Syst. Pervasive Networks*, vol. 9, no. 2, pp. 1–13, 2017, doi: 10.5383/JUSPN.09.02.001.

[12] H. L. Cech, M. Grobmann, and U. R. Krieger, "A fog computing architecture to share sensor data by means of blockchain functionality," *Proc. - 2019 IEEE Int. Conf. Fog Comput. ICFC 2019*, pp. 31–40, 2019, doi: 10.1109/ICFC.2019.00013.

[13] A. K. Shrestha, R. Deters, and J. Vassileva, "User-Controlled Privacy-Preserving User Profile Data Sharing based on Blockchain." arXiv, 2019, doi: 10.48550/ARXIV.1909.05028.

[14] K. Biswas, V. Muthukkumarasamy, and W. L. Tan, "Blockchain based wine supply chain traceability system," in *Future Technologies Conference (FTC) 2017*, 2017, pp. 56–62.

[15] A. Ismailisufi, T. Popović, N. Gligorić, S. Radonjic, and S. Šandi, "A Private Blockchain Implementation Using Multichain Open Source Platform," in *2020 24th International Conference on Information Technology (IT)*, 2020, pp. 1–4, doi: 10.1109/IT48810.2020.9070689.

[16] R. Chinnaiyan and S. Balachandar, "Reliable Administration Framework of Drones and IoT Sensors in Agriculture Farmstead using Blockchain and Smart Contracts," *Proc. 2020 2nd Int. Conf. Big Data Eng. Technol.*, 2020, doi: 10.1145/3378904.

[17] F. A. Al-Zahrani, "Subscription-Based Data-Sharing Model Using Blockchain and Data as a Service," *IEEE Access*, vol. 8, pp. 115966–115981, 2020, doi: 10.1109/ACCESS.2020.3002823.

[18] H. F. Al-Turkistani and N. K. AlSa'awi, "Poster: Combination of Blockchains to Secure Smart Home Internet of Things," in *2020 First International Conference of Smart Systems and Emerging Technologies (SMARTTECH)*, 2020, pp. 261–262, doi: 10.1109/SMART-TECH49988.2020.00069.

[19] M. A. Ferrag and L. Maglaras, "DeepCoin: A Novel Deep Learning and Blockchain-Based Energy Exchange Framework for Smart Grids," *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1285–1297, 2020, doi: 10.1109/TEM.2019.2922936.

[20] E. Fernando, Meyliana, H. L. H. Spits Warnars, and E. Abdurachman, "Blockchain technology for tracing drug with a multichain platform: Simulation method," *Adv. Sci. Technol. Eng. Syst.*, vol. 6, no. 1, pp. 765–769, 2021, doi: 10.25046/AJ060184.

[21] S. Nirmala, M. Sathya, M. Senthilmurugan, R. Chinnaiyan, and S. Alex, "Blockchain based Secured Framework for Road Traffic Management using Fog Computing."

[22] A. Bhattacharjya, K. Kozdrój, G. Bazydło, and R. Wisniewski, "Trusted and Secure Blockchain-Based Architecture for Internet-of-Medical-Things," *Electron.*, vol. 11, no. 16, 2022, doi: 10.3390/electronics11162560.