© 2015 IJSRSET | Volume 1 | Issue 5 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Secure Authentication Protocol for Internet of Things(IoT) Applications based on Blockchain Technology

Voore Subba Rao, Dr. S.K. Tyagi
Department of CSE, Chaudhary Charan Singh University, Meerut, Uttar Pradesh, India

ABSTRACT

Internet of Things is evolving as an autonomous system connecting every possible object to an overarching network. However, the objects in the network are heterogeneous and resource constraint; and, security is one among the primary challenges. Existing security mechanisms are usually relying on a centralized security server; primarily, devices authenticate themselves from a trusted third party. If the server fails to function, then the security mechanism would halt the network. But, Blockchain technology with its decentralization property gives a reliable distributed solution to the single-point failure problem. This paper proposes an authentication mechanism in a permissioned network blockchain for device authentication. The security scheme is tested through a formal authentication tool, Scyther, to verify its authentication properties. The security framework adopted for the proposed system is Hyperledger Fabric blockchain platform. The proposed security scheme over the decentralized network is secure and suitable for IoT applications, which withstands many existing attacks related to authentication.

Keywords - Blockchain, Internet of Things, Authentication, Security, De- centralized, autonomous system

I. INTRODUCTION

With the rapid growth of technology, the number of Internet of Things(IoT) devices has reached to 26.66 billion in 2019. The expected number would reach 30 billion by generating USD 7.1 trillion market economy in 2020. However, IoT is facing security as one of the significant challenges. IoT devices usually enabled with limited memory and processing resources, thereby makes the design of security mechanism more challenging. IoT has six significant security aspects, such as Trustless Environment, Access Control, Data Security, Device Security, Key Management.

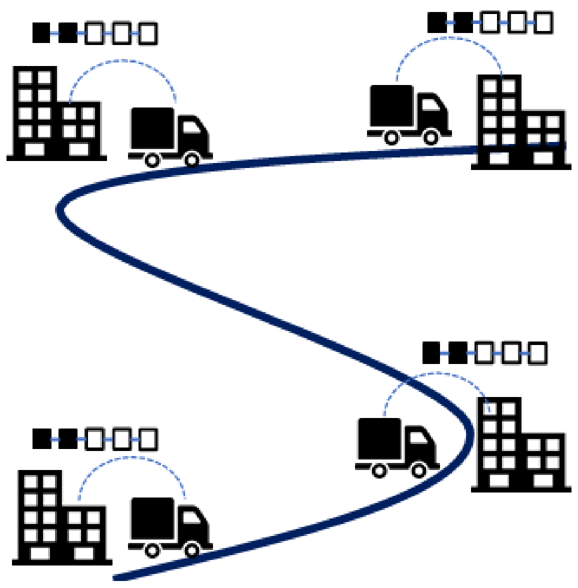


Fig1: Blockchain enabled IoT scenario

Authentication is the essential security requirement for both Data and Device security, where an IoT communication system requires proper authentication mechanism for all the components, namely a user, device, data origin and communication session. The security mechanism must authenticate all the components and withstand various known practical authentication attacks such as Man-in-the Middle(MitM), replay, preplay, and many more. Many IoT organizations consist of various heterogeneous devices which also need to be scalable to support the network growth. The existing authentication mechanisms rely on some centralized, trusted third party or an authentication server. However, the malfunction or failure of the central server would halt the system. But, A decentralized security mechanism Blockchain would solve the problem. It is based on a tamper-proof distributed ledger that does not require a central authority, and every node maintains a local copy of the global data. A blockchain node attains consistency among the local copies by some consensus algorithm and stores the transaction in the temper-proof blockchain. The paper proposed an authentication mechanism in a permissioned network based blockchain platform, Hyperledger Fabric.

The rest of the paper is organized as follows. Section II presents related works of IoT security. Then, Section III analyses the existing security mechanism and explains the improved security features of our scheme. Then, Section IV provides the proposed authentication security mechanism to mitigate threats in the IoT. Next, Section V presents the analysis of proposed security mechanism with the tests of authentication properties by a formal security testing tool Scyther. Subsequent Section VI shows the blockchain implementation of the proposed scheme. Finally, Section VII concludes the paper, followed by References.

II. RELATED WORK

Blockchain is evolving as an alternative security solution for IoT enabled applications. The recent development in Blockchain technology has attracted many researchers to work on IoT security using blockchain. Below are the recent works attempted in the course of blockchain in IoT security.

Associating blockchain security framework in IoT is a challenging task. To identify the security requirements for a system based on three characteristics: a dynamic environment, heterogeneity of system, resource constraints. It analyses the security problems over six primary elements: user, platform, cloud, network, service, and attacker. There are many researches based on security framework.

Rekleitis et.al.[1] proposed firstly about security and privacy policies control in RFID systems. Secondly, proposed a secure and privacy-preserving tag management protocol that support tag authentication, delegation, and ownership transfer and also require minimal hardware and computational requirements for implementation.

Alex Norta et.al.[2] proposed a blockchain technology framework for setup-lifecycle of cross-organizational business-process for decentralized autonomous organizations (DAO).

Wright et.al.[3] proposed a sub-set of Blockchain technology i.e.Lex Cryptographia. The blockchain technology is a decentralized system to store and manage information. Blockchain technology support for smart contracts that can control over the Internet. The authors proposed a new subset Lex Cryptographia rules the autonomos organizations through smart contracts more efficiently than Blockchain technology.

Herbert et.al.[4] proposed cryptocurrency blockchain technology to controlsoftware piracy is a method of decentralized peer-to-peer cost effective method.

I. Alqassem et.al.[5] proposed a requirement engineering framework for implement security and privacy feature for the Internet of Things for its complex nature.

X Huang, P Craig et.al.[6] proposed a framework for device authentication and access control problem by develop a prototype security framework. This security mechanism includes body IoT, home IoT, and hotel IoT with authentication mechanism, access control and risk indicators for providing security and also for solving feasible solutions.

Kim, Young-Pil et.al.[7] proposed an authentication mechanism in IoT In this paper, that is a dynamic and energy-aware authentication scheme for the Internet of Things (DAoT). DAoT uses a feedback control scheme to dynamically select an energy-efficient authentication policy. With DAoT, IoT devices with limited resources can be safely interconnected because DAoT finds and adopts the best cost-effective authentication mechanism.

Gubbi et al.[8] present a Cloud-centric view for global deployment of the Internet of Things using Aneka computing platform. It attempted for authentication in between various layers and terminal IoT nodes. The mechanism uses hash functionality. The scheme primarily deals with one-way authentication from the IoT layer to end nodes but not the reverse. And, there is no practical proof to support the security measure.

H. J. Lee et al.[9] suggested an improved mechanism for access control and authentication in IoT. The paper proposed an Elliptic Curve Cryptography (ECC) based secure key establishment protocol for authentication and a Role-based access control mechanism in IoT applications. Nonetheless, the paper did not provide any proof for security features, and sensor nodes suffer from high communication latency.

Ricardo Neisse et al.[10] provided a model-based Security Toolkit, where it devised a public key technique based identity authentication model for the IoT devices. The timestamp is incorporated in the authentication mechanism to mitigate Man-in-the-Middle attack. The process of authentication follows three sequential steps, i.e., secret key generation, device identity establishment and implementation of granting access control. Even though the security mechanism is not secure from DoS attacks, it reduces the risk associated with it by granting one device at a time.

T. Bose et al.[11] devised a mechanism to secure channel establishment. Primary, it serves two objectives, viz. privacy risk assessment and optimizing the secure transmission based on that assessment. It controls the access and privacy from the obtained sensors information. Although it improves security with minimal resource consumption, it assumes for a single security case.

Ning et al.[12] proposed an access control and authentication mechanism for the perception layer of IoT. This paper uses

Elliptical Curve Cryptography based session key for mutual authentication between a user and IoT nodes. Nonetheless, it only considers authentication issues in perception layer of IoT and does not deal with access control policy between other devices.

III. CRYPTANALYSIS OF EXISTING WORKS

Alex Norta et.al.[2] proposed a blockchain technology framework for setup-lifecycle of cross-organizational business-process for decentralized autonomous organizations (DAO). The CPN tools used to explore P2P collaboration model for setup-lifecycle for graphical notations. Authors plan to develop blockchain-technology for Internet-of-Things. In future work improve blockchain technology for effective management of trust, privacy and security in cross-organizational cyber-physical system collaboration.

Fig2 is P2P collaboration model is business network model (BNM) is creation of smart-contracting collaboration is tree base model using eSourcing framework. In Fig1(a) service offers that match with service types in BNM. In Fig1(b) shows smart contracting by eSourcing Markup Language (eSML).

Interrogatives Who for contracting parties with their resources, Where to note business and legal activity, and What notice match process-views.

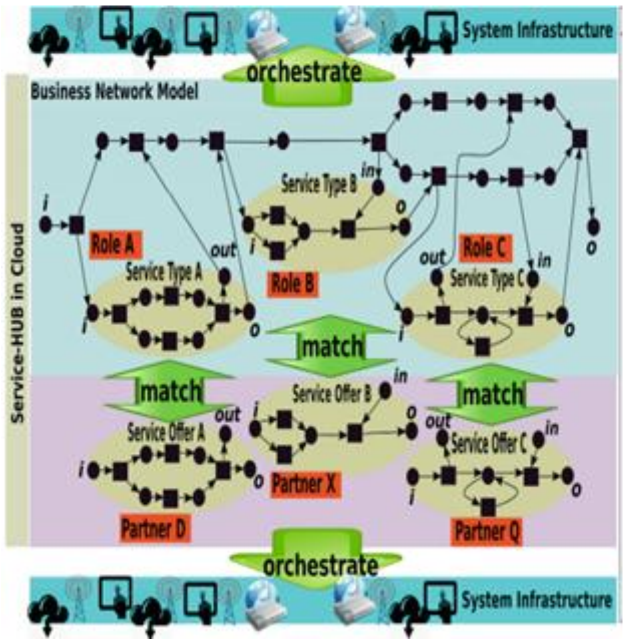
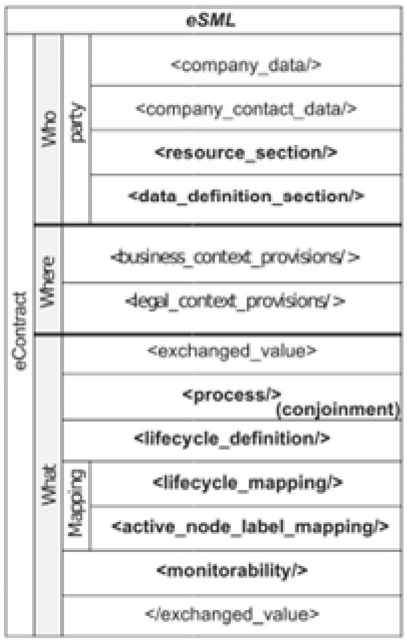


Fig.2 P2P-collaboration using the eSourcing framework. (a) service offers that match with service types in BNM



eSML			
eContract	Who	party	<company_data/>
			<company_contact_data/>
			<resource_section/>
			<data_definition_section/>
	Where		<business_context_provisions/>
			<legal_context_provisions/>
	What		<exchanged_value>
			<process/> **(conjoinment)**
			<lifecycle_definition/>
		Mapping	**<lifecycle_mapping/>**
			<active_node_label_mapping/>
			<monitorability/>
			</exchanged_value>

(b) shows smart contracting by eSourcing Markup Language (eSML)

Fig2 is P2P collaboration model is business network model (BNM) is creation of smart-contracting collaboration is tree base model using eSourcing framework.

In Fig2(a) service offers that match with service types in BNM. In Fig2(b) shows smart contracting by eSourcing Markup Language (eSML). interrogatives Who for contracting parties with their resources, Where to note business and legal activity, and What notice match process-views.

IV. PROPOSED SECURE AUTHENTICATION PROTOCOL

A. System Model

Fig3 shows a typical transaction flow in a blockchain enabled IoT infrastructure. Let us consider an IoT enabled supply-chain system where a vehicle required to transport a critical material whose temperature must be under four degrees centigrade. So, at every intermediate node of the supply chain, various properties of the article, such as temperature, vibration, and humidity, must be submitted to the global ledger.

At the outset, autonomous IoT node senses the properties and prepares the update transaction. It then sends the digitally signed transaction to all the connected peers in the blockchain network. At the receivers end, all the peers verify the signature put by IoT node and execute the transaction. Each executing peer acts as an endorsor and captures the set of Read and Written data called RW Sets. Then they send their RW sets to IoT node. IoT node receives the result status asynchronously and submits those to ordering service after verification of peers' signatures. The ordering process happens across the system in parallel with the transaction submitted by other IoT applications. It collects the transactions into a proposed block and sends to peers for validation. The validator validates the transaction against every endorser. It further checks for the data inconsistency and demarcates the invalid transaction. Next, it forwards the validated block, digitally signed, to committers. At the end of the process, the committer saves the proposed transaction block to the global ledger.

B. Assumptions

The proposed scheme in this paper relies on some plausible assumptions which require the system to work. The following assumptions are considered for the proposed device authentication scheme:

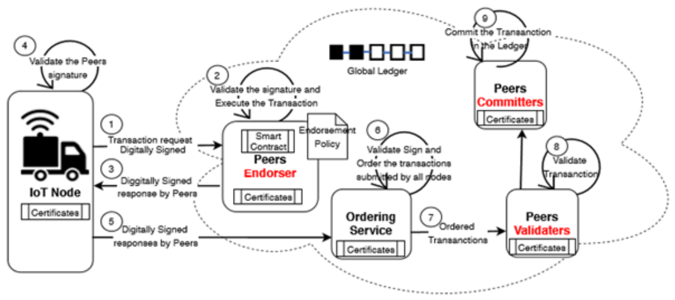

4
Validate the Peers signature
2
Validate the signature and Execute the Transaction
Global Ledger
9
Commit the Transanction in the Ledger
Peers
Committers
Certificates
1
Transaction request Digitally Signed
Smart Contract
Peers
Endorser
Certificates
Endorsement Policy
6
Validate Sign and Order the transactions submitted by all nodes
8
Validate Transanction
3
Diggitally Signed response by Peers
IoT Node
Certificates
5
Digitally Signed responses by Peers
Ordering Service
Certificates
7
Ordered Transanctions
Peers
Validaters
Certificates

Fig3 : A Typical Infrastructure of Blockchain Enabled IoT

1. Every IoT node has its corresponding unique identifica- tion number.
2. Each IoT node has a pairwise public-private key pair. Each Peer node(IoT cluster head) has sufficient storage and processing capabilities to implement smart contract.
3. There may be more than one IoT node connected to Peer.
4. All nodes in the network trusts the node manager for initial registration.
5. Each IoT node knows the public key of its associated Peers.

C. Threat Model

The proposed scheme considers Canett-Krawczyk[13] and Delov-Yao[14] threat models in a Blockchain-enabled IoT system. Devices in a supply chain management systems are IoT enabled and forms a permissioned network. The IoT enabled vehicle carries various packages which itself IoT enabled and collects various essential features of the package. The package characteristics is agreed and contracts are established between parties in the supply chain system.

The inappropriate carrier characteristics may violate the contract and thus prices may change. The twentieth century is enabled with so many modern network tool and knowledge of adversary is also increasing. A malicious attacker can eavesdrop and set wrong data for a device and package, thereby changing the pricing policy.

An attacker possibly launch various related authentication attacks, for example, Reflection, Replay, Certificate Manipulation, Man-in-the-Middle, Typing attacks, Preplay, and Denial of Service, which are serious threat to the system. An adversary is able to eavesdrop on all messages sent in insecure channel. He can alter, reroute all messages sent; generate and insert entirely new malicious messages.

Attacker can also pretend to be a legitimate protocol participant (dishonest user) or external intruder or a combination of both. He can obtain any sufficiently old previously run session key start any number of parallel protocol runs.

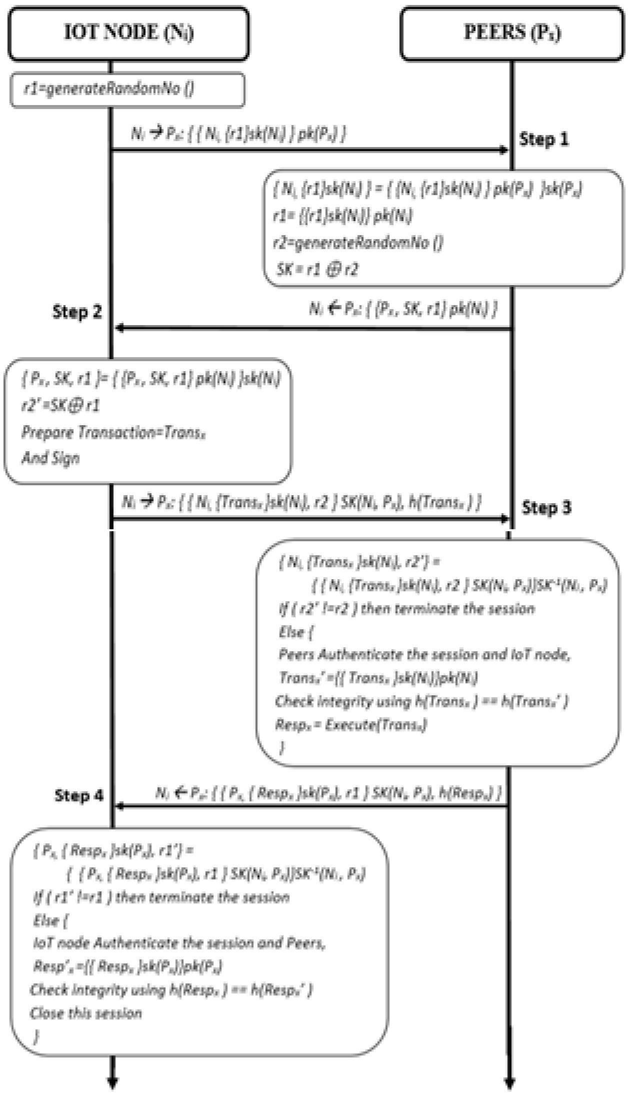
Table I: Notations Used in Security Protocol

Acronym	Description
N_i	Identity of i^{th} IoT node
P_x	Identity of Peer in blockchain
$r1$	Random number generated by IoT node
$r2$	Random number generated by Peer
SK	Secret Shared Symmetric key
$Trans_x$	Transaction requests from IoT node
$Resp_x$	Read Write result set
$\{M\}pk(X)$	Message M encrypted by the public key of user X
$\{M\}sk(X)$	Message M signed by the private key of user X
$\{M\}SK(X,Y)$	Message M is encrypted by the shared symmetric key of user X and Y
$\{M\}SK^{-1}(X,Y)$	Message M is decrypted by the shared symmetric key of user X and Y
$h(M)$	Message digest of message M using one way hash function h()

D. Authentication Protocol

Fig4 shows the proposed authentication protocol in the blockchain enabled autonomous IoT system. The abbreviations and notations used in the protocol are depicted in Table I. In a typical blockchain network, there are three types of peers: Endorser, Validators, and Committers. Initially, when an IoT node, Ni, wishes to connect any of the peers, Px, it generates a random number, r1, and puts its digital signature on it.

Next, it prepares the message by concatenating the digitally signed random number and its identity. Then, it encrypts the message and sends to the peer, Px. The message is {Ni,{r1}sk(Ni)}pk(Px).



IOT NODE (Ni)
PEERS (Px)
r1=generateRandomNo ()
Ni → Px: { { Ni, {r1}sk(Ni) } pk(Px) }
Step 1
{ Ni, {r1}sk(Ni) } = { {Ni, {r1}sk(Ni) } pk(Px) }sk(Px)
r1= {{r1}sk(Ni)} pk(Ni)
r2=generateRandomNo ()
SK = r1 ⊕ r2
Step 2
Ni ← Px: { {Px, SK, r1} pk(Ni) }
{ Px, SK, r1 }= { {Px, SK, r1} pk(Ni) }sk(Ni)
r2' =SK ⊕ r1
Prepare Transaction=Transx
And Sign
Ni → Px: { { Ni, {Transx }sk(Ni), r2 } SK(Ni, Px), h(Transx) }
Step 3
{ Ni, {Transx }sk(Ni), r2'} =
{ { Ni, {Transx }sk(Ni), r2 } SK(Ni, Px)}SK⁻¹(Ni, Px)
If (r2' !=r2) then terminate the session
Else {
Peers Authenticate the session and IoT node,
Transx' ={{ Transx }sk(Ni)}pk(Ni)
Check integrity using h(Transx) == h(Transx')
Respx = Execute(Transx)
}
Step 4
Ni ← Px: { { Px, { Respx }sk(Px), r1 } SK(Ni, Px), h(Respx) }
{ Px, { Respx }sk(Px), r1'} =
{ { Px, { Respx }sk(Px), r1 } SK(Ni, Px)}SK⁻¹(Ni, Px)
If (r1' !=r1) then terminate the session
Else {
IoT node Authenticate the session and Peers,
Resp'x ={{ Respx }sk(Px)}pk(Px)
Check integrity using h(Respx) == h(Respx')
Close this session
}

Fig 4 - Proposed Mutual Authentication and Key-Agreement Protocol

The peer Px, upon receiving the message from the node Ni, decrypts the message using its secret key. It extracts the identity of sender, Ni, and the signed random number, {r1}sk(Ni). Next, it obtains the random number by using the public key of the sender, i.e., {{r1}sk(Ni)}pk(Ni). Then, it prepares the shared session key, SK, using the X-OR operation on r1 and r2. After that, it prepares a message including its identity Px, the session key SK, and the received random number r1. And, it encrypts the message using the public key of IoT node and sends back to the IoT node. The encrypted message is{Px,SK,r1}pk(Ni).

The IoT node receives the message from all the peers asynchronously; Nonetheless, it decrypts the message using its private key. It extracts the content of the message, {Px,SK,r1}, and identifies the peer by inspecting the identity Px. Further, the node checks the corresponding random number and verifies if it is equal to its sent version. Next, it extracts the peer's random number by applying X-OR operation on SK and r1. Following that, it prepares a transaction and put its digital sign on it, i.e., {Transx}sk(Ni).

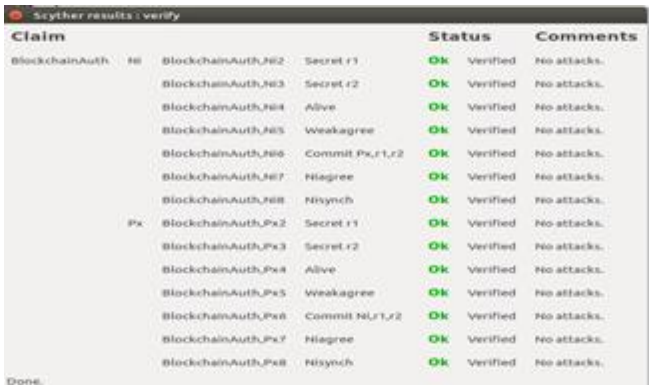
Subsequently, it prepares the message containing its identity, the digital signature and the peer's random number. The message is {Ni,{Transx}sk(Ni),r2}. Then it encrypts the message using the shared session key and symmetric key algorithm rather than the receiver's public key. It also prepares a message digest for the transaction using a one- way hash function and sends it along with the encrypted message. The combined message to be sent is ..
{Ni,{Transx}sk(Ni),r2}SK(Ni,Px),h(Transx)}.

The peer Px receives the message from the node Ni. It decrypts the message using the generated shared secret key SK and obtains the identity of the node, digitally signed transaction and a random number. The peer checks the equality between the received random number r20 and its initial sent version r2. If the equality holds, then it authenticates the IoT node and the communication session. Then, it extracts the transaction using the public key of the node, i.e., Transx = {{Transx}sk(Ni)}pk(Ni). It also verifies the integrity of the transaction by comparing the message digests. The peers(Endorser) executes the requested transaction and records the Read and Write result set. In the end, it prepares the message containing its identity, digitally signed result set, and the node's random number r1. Subsequently, it encrypts the message using the shared secret key SK. It also prepares a message digest for the result set and sends it along with the encrypted message to the IoT node, Ni. The message is {{Px,{Respx}sk(Px),r1}SK(Ni,Px), h(Respx)}.

The IoT node receives the messages from the peer, decrypts it using the shared key, and extracts the content. It checks the equality between its initial random number and the received version from the peer. If equality holds good, then it authenticates the peer and the communication. At this point, both the parties authenticate each other; therefore, mutual authentication establishes. After that, the node verifies the digital signature on the result set. Subsequently, it forwards the result set, signed by peers, to the ordering service.

V. PROTOCOL VERIFICATION USING SCYTHER

The authentication and key-agreement protocol is simulated using a role-based security verification tool Scyther[15]. The tool is a formal method based and tests various authentication propoerties, namely Secrecy, Non-Injective Agreement(NI-Agree), Non-Injective Synchronization(NISynch) and Aliveness. The proposed protocol is written in Security Protocol Description Language (.spdl) and tested for various known authentication attacks.



```
Scyther results : verify
Claim                                                        Status          Comments
BlockchainAuth  Ni  BlockchainAuth,Ni2  Secret r1            Ok   Verified   No attacks.
                    BlockchainAuth,Ni3  Secret r2            Ok   Verified   No attacks.
                    BlockchainAuth,Ni4  Alive                Ok   Verified   No attacks.
                    BlockchainAuth,Ni5  Weakagree            Ok   Verified   No attacks.
                    BlockchainAuth,Ni6  Commit Px,r1,r2      Ok   Verified   No attacks.
                    BlockchainAuth,Ni7  Niagree              Ok   Verified   No attacks.
                    BlockchainAuth,Ni8  Nisynch              Ok   Verified   No attacks.
                Px  BlockchainAuth,Px2  Secret r1            Ok   Verified   No attacks.
                    BlockchainAuth,Px3  Secret r2            Ok   Verified   No attacks.
                    BlockchainAuth,Px4  Alive                Ok   Verified   No attacks.
                    BlockchainAuth,Px5  Weakagree            Ok   Verified   No attacks.
                    BlockchainAuth,Px6  Commit Ni,r1,r2      Ok   Verified   No attacks.
                    BlockchainAuth,Px7  Niagree              Ok   Verified   No attacks.
                    BlockchainAuth,Px8  Nisynch              Ok   Verified   No attacks.
Done.
```

Fig5 - Syther Simulation Result of Mutual Authentication and Key-Agreement Protocol

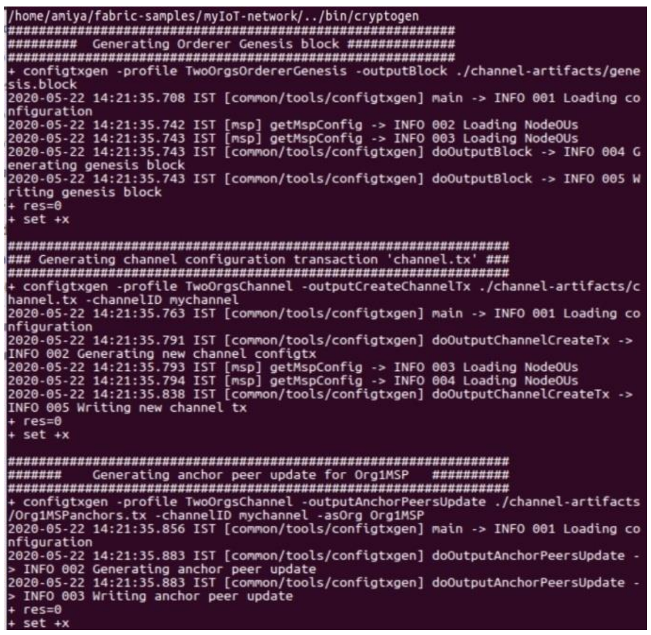
The tool executes for fifty runs per agents, i.e., there are fifty instances of the protocol running in multiple nodes. The simulation result is shown in Figure 6. Ni is simulated for the IoT node, and Px is for peer nodes in the blockchain network. The secrecy property of the randoms r1 and r2 is preserved. Thus, the generated shared secret key is secure, and the protocol withstands various impersonation attacks, for example, replay, reflection, and typing attacks. The aliveness property implies the nodes are active in the current session and not any past communication session. NI-Agree means that the contents of the received messages correspond to the sent messages, i.e., the order of the contents is preserved and the scheme is resistant to attacks related to typing attack. The NI-synch properties in the simulation result convey that the messages communicated in the protocol maintains its order and thus attacks manipulating the order of messages is not possible. The simulation test results showed that the proposed scheme is secure and withstands all known attacks related to authentication.

The paper used the cryptogen tool to generate secret cryptographic documents, an x509 certificate based on standard PKI. The tool requires a configuration file, "crypto-config.yaml". It primarily contains a unique root certificate and public-private key-pair for peers. The orderer genesis block is generated, and the channel is configured. Then anchor peers are generated who can communicate to orderer service. Figure 7 shows the snapshots of the genesis block, channel and anchor peers creations. Next, the channel is created, and peers joined the channel. Subsequently, the chaincode is installed and instantiated in peers. Figure 8 and 9 show the snapshot of the peer joining the created channel and chaincode installation on the peers.

VI. HYPERLEDGER FABRIC BLOCKCHAIN PLATFORM

We simulated the Hyperledger Fabric platform in a Ubuntu virtual set up. It is an implementation of blockchain technology developed by Linux Foundation's Hyperledger Project. The initial setup required some prerequisites, such as cURL, Docker, Docker Compose, Golang programming language, Nodejs lang version 8.x, and Python 2.7. Fig3 shows a typical Fabric platform that contains three types of

Peers: Endorser, Validators, and Committers. Peers communicate through private channels. A channel provides privacy for a group of peers. Each peer installs the smart-contract, called Chaincode in Fabric, to provide interface and functionality to the end-users. The docker container of Hyperledger Fabric is publicly available for test.



```
/home/amiya/fabric-samples/myIoT-network/../bin/cryptogen
##########################################################
#########  Generating Orderer Genesis block ##############
##########################################################
+ configtxgen -profile TwoOrgsOrdererGenesis -outputBlock ./channel-artifacts/gene
sis.block
2020-05-22 14:21:35.708 IST [common/tools/configtxgen] main -> INFO 001 Loading co
nfiguration
2020-05-22 14:21:35.742 IST [msp] getMspConfig -> INFO 002 Loading NodeOUs
2020-05-22 14:21:35.743 IST [msp] getMspConfig -> INFO 003 Loading NodeOUs
2020-05-22 14:21:35.743 IST [common/tools/configtxgen] doOutputBlock -> INFO 004 G
enerating genesis block
2020-05-22 14:21:35.743 IST [common/tools/configtxgen] doOutputBlock -> INFO 005 W
riting genesis block
+ res=0
+ set +x

#################################################################
### Generating channel configuration transaction 'channel.tx' ###
#################################################################
+ configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./channel-artifacts/c
hannel.tx -channelID mychannel
2020-05-22 14:21:35.763 IST [common/tools/configtxgen] main -> INFO 001 Loading co
nfiguration
2020-05-22 14:21:35.791 IST [common/tools/configtxgen] doOutputChannelCreateTx ->
INFO 002 Generating new channel configtx
2020-05-22 14:21:35.793 IST [msp] getMspConfig -> INFO 003 Loading NodeOUs
2020-05-22 14:21:35.794 IST [msp] getMspConfig -> INFO 004 Loading NodeOUs
2020-05-22 14:21:35.838 IST [common/tools/configtxgen] doOutputChannelCreateTx ->
INFO 005 Writing new channel tx
+ res=0
+ set +x

#################################################################
#######    Generating anchor peer update for Org1MSP   ##########
#################################################################
+ configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./channel-artifacts
/Org1MSPanchors.tx -channelID mychannel -asOrg Org1MSP
2020-05-22 14:21:35.856 IST [common/tools/configtxgen] main -> INFO 001 Loading co
nfiguration
2020-05-22 14:21:35.883 IST [common/tools/configtxgen] doOutputAnchorPeersUpdate -
> INFO 002 Generating anchor peer update
2020-05-22 14:21:35.883 IST [common/tools/configtxgen] doOutputAnchorPeersUpdate -
> INFO 003 Writing anchor peer update
+ res=0
+ set +x
```

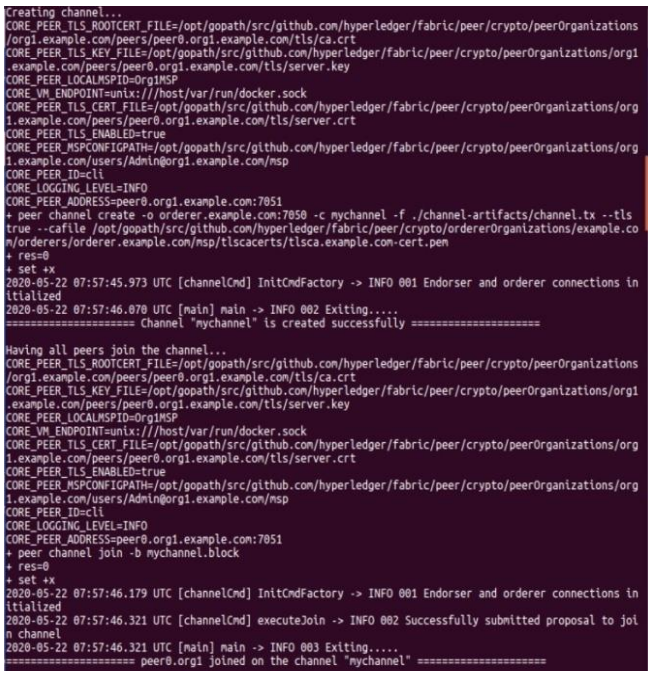
Fig6: Orderer genesis block creation, Channel configuration and Anchor peer generation

A set of steps followed for setting up a blockchain are the following:

1) Configure and start ordering service.
2) Configure and start peer nodes.
3) Install a chaincode in each peer.
4) Create channels.
5) Join channels to peers.
6) Instantiate chaincode in the channel

VII. CONCLUSION

The paper considered an IoT enabled scenario where blockchain technology is used to achieve a decentralised security mechanism. An IoT node needs to be adequately authenticated before making any communications with peers of blockchain infrastructure. The paper proposed a secure authentication and Key-Agreement protocol for IoT node and Peers. The protocol has used public key and generated a shared symmetric key which further secure communication It provided mutual authentication for the two-party communication. Protocol tested by a formal security tool Scyther. It is observed that the proposed protocol is robust enough to withstand all the known attacks related to authentication, such as Replay, Preplay, Typing Attacks and so on. The implemented blockchain technology, i.e., Hyperledger Fabric, provided better IoT enabled system. It also supported the scalability of the IoT devices in the network. Hence, the proposed scheme is a scalable and secure scheme for device authentication in a blockchain-enabled IoT environment.



```
Creating channel...
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations
/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1
.example.com/peers/peer0.org1.example.com/tls/server.key
CORE_PEER_LOCALMSPID=Org1MSP
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org
1.example.com/peers/peer0.org1.example.com/tls/server.crt
CORE_PEER_TLS_ENABLED=true
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org
1.example.com/users/Admin@org1.example.com/msp
CORE_PEER_ID=cli
CORE_LOGGING_LEVEL=INFO
CORE_PEER_ADDRESS=peer0.org1.example.com:7051
+ peer channel create -o orderer.example.com:7050 -c mychannel -f ./channel-artifacts/channel.tx --tls
true --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.co
m/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem
+ res=0
+ set +x
2020-05-22 07:57:45.973 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections in
itialized
2020-05-22 07:57:46.070 UTC [main] main -> INFO 002 Exiting.....
===================== Channel "mychannel" is created successfully =====================

Having all peers join the channel...
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations
/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1
.example.com/peers/peer0.org1.example.com/tls/server.key
CORE_PEER_LOCALMSPID=Org1MSP
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org
1.example.com/peers/peer0.org1.example.com/tls/server.crt
CORE_PEER_TLS_ENABLED=true
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org
1.example.com/users/Admin@org1.example.com/msp
CORE_PEER_ID=cli
CORE_LOGGING_LEVEL=INFO
CORE_PEER_ADDRESS=peer0.org1.example.com:7051
+ peer channel join -b mychannel.block
+ res=0
+ set +x
2020-05-22 07:57:46.179 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections in
itialized
2020-05-22 07:57:46.321 UTC [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to joi
n channel
2020-05-22 07:57:46.321 UTC [main] main -> INFO 003 Exiting.....
===================== peer0.org1 joined on the channel "mychannel" =====================
```

Fig7: Chaincode installation on peers

VIII. ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to my Supervisor for valuable guidance and support in completing my research paper. I would like to extend my gratitude to the Principal for providing me with facilities are required.

IX. REFERENCES

[1] . Rekleitis, Evangelos, Panagiotis Rizomiliotis, and Stefanos Gritzalis. "How to protect security and privacy in the IoT: a policy-based RFID tag

management protocol." Security and Communication Networks 7.12 (2014): 2669-2683.

[2] . Norta, Alex. "Creation of smart-contracting collaborations for decentralized autonomous organizations." International Conference on Business Informatics Research. Springer, Cham, 2015.

[3] . Wright, Aaron, and Primavera De Filippi. "Decentralized blockchain technology and the rise of lex cryptographia." Available at SSRN 2580664 (2015).

[4] . Herbert, Jeff, and Alan Litchfield. "A novel method for decentralised peer-to-peer software license validation using cryptocurrency blockchain technology." Proceedings of the 38th Australasian Computer Science Conference (ACSC 2015). Vol. 27. 2015.

[5] . Alqassem. Privacy and security requirements framework for the internet of things (iot). In International Conference on Software Engineering (ICSE) Companion India, pages 739–741, May-June 2014.

[6] . Huang, P. Craig, H. Lin, and Z. Yan. Seciot: a security framework for the internet of things. In Security and communication networks, May 2015.

[7] . Kim, Young-Pil, Seehwan Yoo, and Chuck Yoo. "DAoT: Dynamic and energy-aware authentication for smart home appliances in Internet of Things." 2015 IEEE International Conference on Consumer Electronics (ICCE). IEEE, 2015.

[8] . Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7):1645–1660, 2013. Including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services & Cloud Computing and Scientific Applications — Big Data, Scalable Analytics, and Beyond.

[9] . Ndibanje B., Lee H. J., and Lee S. G. Security analysis and improvements of authentication and access control in the internet of things. Sensors, 14(8):14786–14805, 2014.

[10] . Ricardo Neisse, Gary Steri, Igor Nai Fovino, and Gianmarco Baldini. Seckit: A model-based security toolkit for the internet of things. Computers & Security, 54:60 – 76, 2015. Secure Information Reuse and Integration & Availability, Reliability and Security 2014.

[11] . T. Bose, S. Bandyopadhyay, A. Ukil, A. Bhattacharyya, and A. Pal. Why not keep your personal data secure yet private in iot?: Our lightweight approach. In 2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), pages 1–6, April 2015.

[12] . Ning YE, Yan Zhu, Ru-chuan WANG, Reza Malekian, and Lin Qiaomin. An efficient authentication and access control scheme for perception layer of internet of things. 8, 07 2014.

[13] . Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Birgit Pfitzmann, editor, Advances in Cryptology — EUROCRYPT 2001, pages 453–474, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[14] . D. Dolev and A. C. Yao. On the security of public key protocols. In 22nd Annual Symposium on Foundations of Computer Science (sfcs 1981), pages 350–357, 1981.

[15] . Sjouke Mauw Cas Cremers. Operational Semantics and Verification of Security Protocols. Springer-Verlag Berlin Heidelberg, 1 edition, 2012.