

# Final review on Mini Project

## “Attribute Based Access Control using Blockchain”



By:

Sujith Reddy M	321910304034
Sathish Kumar H S	321910304035
Kartik Patil	321910304055
Damera Dinesh	321910304060

Under the Guidance of  
Rajesh S M  
Assistant Professor  
Department of CSE  
GITAM University Bengaluru



# CONTENTS

- ☐ Abstract
- ☐ Introduction
- ☐ Literature Survey
- ☐ Problem Definition
- ☐ Objectives
- ☐ Solution Strategy
- ☐ Methodology
- ☐ Implementation
- ☐ Project Structure Flow
- ☐ Execution calls
- ☐ Working Environment
- ☐ Testing
- ☐ Experimental Results
- ☐ Conclusion
- ☐ Future work
- ☐ References



## Abstract :-

- A large amount of data is being created every day on the internet. This includes our personal data. Access to this personal data should be regulated properly. Eg: Internet of Things (IoT) devices produce a lot of private data.
- Traditional centralized access control systems lead to shortcomings of a single point of failure, low overall system efficiency, and ethical and privacy issues.
- To overcome such challenges, an attribute-based access control model using blockchain.
- ABAC grants access based on the attributes presented by the target.



## Introduction :-

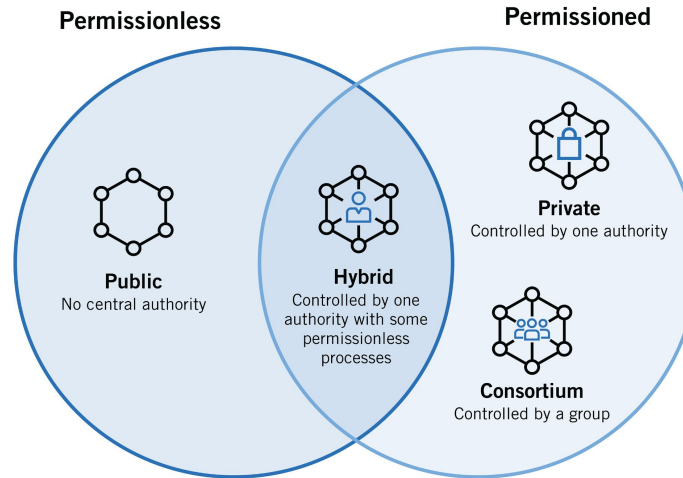
**Access control** is a security technique that regulates who or what can view or use resources in a computing environment. It is a fundamental concept in security that minimizes risk to the business or organization. Two types of access control - Physical and Logical access control.

Physical Access Control -> Campuses, Buildings, Rooms

Logical Access Control -> Types

- Mandatory Access Control
- Discretionary Access Control
- Role Based Access Control
- Attribute Based Access Control
- Identity Based Access Control

**Blockchain** is a list of records called blocks that store data publicly and in chronological order. The information is encrypted using cryptography to ensure that the privacy of the user is not compromised and data cannot be altered. Blockchain is immutable, decentralized, ensures integrity, shared, consistency.



**Fig 1 [1]: Types of Blockchain**

---

**Kotlin** is a new open-source programming language like Java, JavaScript, Python etc. It is a high level strongly statically typed language that combines functional and technical part in a same place.

1. Easy Language
2. Very Concise
3. Runtime and Performance
4. Interoperability
5. Brand New

**JSON (JavaScript Object Notation)** is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays (or other serializable values). It is a common data format with diverse uses in electronic data interchange, including that of web applications with servers.

## Literature Survey :-

Authors Name, Journal Name, Vol., Year, Page	Title of the Paper	Inference	Research Gap	Relevance with the present work
W. Liang, M. Tang, J. Long, X. Peng, J. Xu, and K.-C. Li, “A secure fabric blockchain-based data transmission technique for industrial Internet-of-Things,” IEEE Trans. Ind. Inf., vol. 15, no. 6, pp. 3582–3592, Jun. 2019	A Secure Fabric Blockchain-Based Data Transmission Technique for Industrial Internet-of-Things	Proposed a secure Fabric-based data transmission technique which solved the problems of low security, high management cost, and difficulty in supervision.	Little or no evidence to address the research problem of have low security, high management cost of the trading center, and big difficulty in supervision	Proposes a secure Fabric blockchain-based data transmission technique for industrial IoT. This technique uses the blockchain mechanism.

[2] Table 1 : Literature Survey




Authors Name, Journal Name, Vol., Year, Page	Title of the Paper	Inference	Research Gap	Relevance with the present work
O. Novo, “Blockchain meets IoT: An architecture for scalable access management in IoT,” IEEE Internet Things J., vol. 5, no. 2, pp. 1184–1195, Apr. 2018.	Blockchain Meets IoT: An Architecture for Scalable Access Management in IoT	A centralized access control scheme for the IoT devices was proposed. The scheme used a single smart contract to reduce the communication cost between nodes. It has six advantages, which are Mobility, Accessibility, Concurrency, Lightweight, Scalability and Transparency	As access management technologies exist in IoT, they are based on centralized models which introduce a new variety of technical limitations to manage them globally.	Proposed a new architecture for arbitrating roles and permissions in IoT. The new architecture is a fully distributed access control system for IoT based on blockchain technology.

**[3] Table 2 : Literature Survey**



Authors Name, Journal Name, Vol., Year, Page	Title of the Paper	Inference	Research Gap	Relevance with the present work
Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, “Smart contract based access control for the Internet of Things,” IEEE Internet Things J., vol. 6, no. 2, pp. 1594–1605, Apr. 2019.	Smart Contract-Based Access Control for the Internet of Things.	ACC implemented policy-based authorization by checking the behavior of objects. JC was used to judge the wrong behavior and return the corresponding punishment. RC was used to register the above two smart contracts and provide update, delete, and other operations.	The Practices deviate from research findings. Changing smart contract processes is almost impossible, any error in the code can be time-consuming and expensive to correct. Also, the acc is not the efficient access control mechanism.	Integrating a smart contract-based framework, which consists of multiple access control contracts (ACCs), one judge contract (JC), and one register contract (RC), to achieve distributed and trustworthy access control for IoT systems.

[4] Table 3 : Literature Survey



Authors Name, Journal Name, Vol., Year, Page	Title of the Paper	Inference	Research Gap	Relevance with the present work
G. Papadodimas, G. Palaiokrassas, A. Litke, and T. Varvarigou, “Implementation of smart contracts for blockchain based IoT applications,” in Proc. 9th Int. Conf. Netw. Future (NOF), Nov. 2018, pp. 60–67.	Implementation of smart contracts for blockchain based IoT applications	A distributed application (DAPP) based on Ethereum was proposed. It combines the SaaS business model with blockchain, which was used to buy and sell sensor data.	A distinction in research methods is needed to have new insights or to avoid ambiguous findings for these Sensing-as-a-Service (S2aaS) business models combined with blockchain.	A decentralized application (DApp) based on blockchain technology for sharing Internet of Things (IoT) sensors’ data, and demonstrating various challenges addressed during the development process. This application combines blockchain technology with IoT and operates through smart contracts that are executed on the Ethereum blockchain.

[5] Table 4 : Literature Survey



## Problem Definition :-

- With the distributed deployment of devices and their large number and scale, the access control of device resources is facing significant challenges.
- The access control technology is an essential means to protect resources, widely used in various systems and environments. The centralized designs have disadvantages.
- Attributed-based access control (ABAC) is a logical access control model, which controls the access between subjects and objects, according to the attributes of entries, operations and related environments.



## Objectives :-

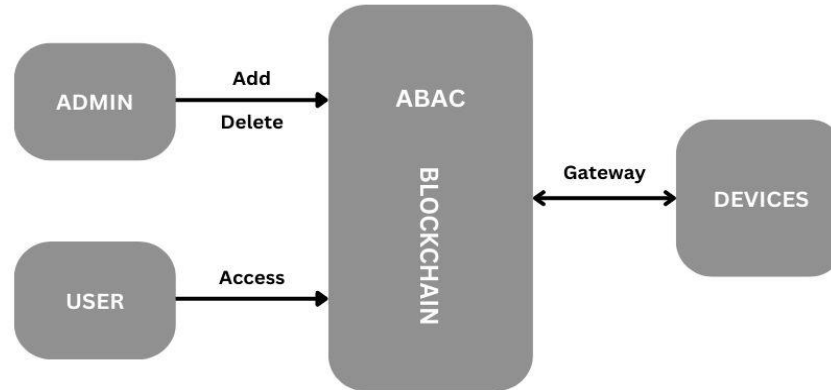
- Using blockchain, we implement ABAC logical access control.
- Blockchain technology offers features similar to decentralization, high confidence, and tamper-resistance, which are advantages to working resource consumption, scalability and trust issues – all of which are challenges for access control by traditional mechanisms.



## Solution Strategy :-

- We propose a blockchain-based access control system and describe its workflow and architecture in detail. The system uses distributed architecture to separate users and devices, and implemented the dynamic management of permissions to support efficient access.
- We implement the ABAC model, ABAC-Policy which contains Attribute of Subject, Attribute of Object, Attribute Environment, and Attribute Permission.

# Methodology



**Fig 2: ABAC Integrated With Blockchain**



## Implementation :-

We are trying to implement 3 algorithms which are as follows:

1. `checkPolicy()`
2. `addPolicy()`
3. `deletePolicy()`



## Implementation :-

P - Policy

AS - Attribute Of Subject

AO - Attribute Of Object

AP - Attribute Of Permission

AE - Attribute Of Environment

$$P = \{ AS, AO, AP, AE \}$$
$$AS = \{ userId, role, group \}$$
$$AO = \{ deviceId \}$$
$$AP = \{ 1, allow$$
$$0, deny \}$$
$$AE = \{ createTime, endTime \}$$



## Implementation :-

```
checkPolicy() : boolean
input -> ABACP
isOK = True
for item in AS do:
    if item != < userId,role, group > then
        isOK = False
    end if
end for
if AO.deviceID != <deviceId> then
    isOK = False
end if
```

```
if val(AP) != 1 or val(AP) != 0
    isOK = False
end if
for item in AE do:
    if item != < createTime, endTime > then
        isOK = False
    end if
end for
return isOK
```



## Implementation :-

```
addPolicy(): Null | Error
Input -> ABACP
if CheckPolicy(ABACP) == False
    return Error("Bad Policy")
end if
Id <- SHA256(ABACP.AS + ABACP.AO)
err <- addABACP(Id, ABACP)
if err != null then
    return Error(err.Text)
end if
return Null
```



## Implementation :-

```
deletePolicy() : Null | Error  
Input -> AS,AO  
Id <- Sha256(AS + AO)  
err <- GetABACP(Id)  
if err! = Null then  
    return Error(err.Text)  
end if  
deleteABACP(Id)  
return Null
```

## Project Flow Structure:-

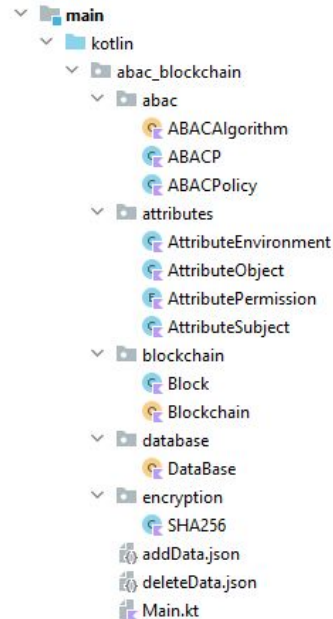


Fig 3: Project Structure

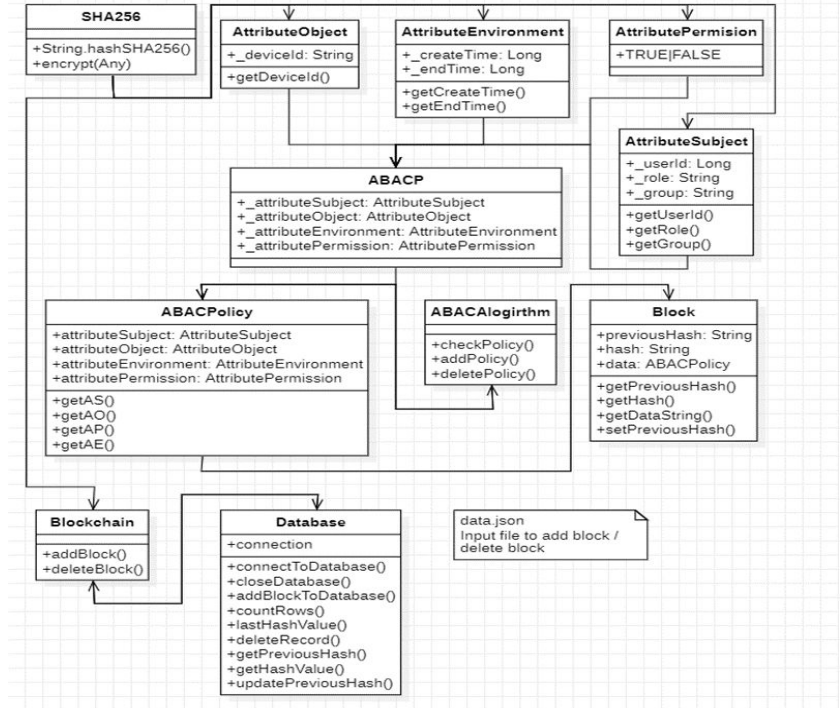


Fig 4: Class diagram ABAC with Blockchain

## Execution calls:-

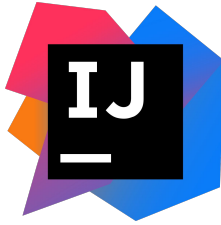
```

main() (abac_blockchain)
  ↳ JSONObject.JSONObject(String) (org.json)
  ↳ JSONObject.getString(String)(8 usages) (org.json)
  ↳ JSONObject.getJSONObject(String)(5 usages) (org.json)
  ↳ JSONObject.getInt(String)(2 usages) (org.json)
  ↳ Locale.getDefault() (java.util)
  ↳ JSONObject.getLong(String)(2 usages) (org.json)
  ↳ ABACAlgorithm.addPolicy(ABACP) (abac_blockchain.abac)
    ↳ Blockchain.addBlock(ABACPolicy) (abac_blockchain.blockchain)
      ↳ SHA256 (abac_blockchain.encryption)
      ↳ SHA256.encrypt(Any?) (abac_blockchain.encryption)
      ↳ ABACPolicy.getASQ(3 usages) (abac_blockchain.abac)
      ↳ AttributeSubject.getUserId() (abac_blockchain.attributes)
      ↳ AttributeSubject.getRole() (abac_blockchain.attributes)
      ↳ AttributeSubject.getGroup() (abac_blockchain.attributes)
      ↳ ABACPolicy.getAO() (abac_blockchain.abac)
      ↳ AttributeObject.getDeviceld() (abac_blockchain.attributes)
      ↳ DataBase.countRows() (abac_blockchain.database)
      ↳ DataBase.lastHashValue() (abac_blockchain.database)
      ↳ DataBase.addBlockToDatabase(Block) (abac_blockchain.database)
    ↳ ABACAlgorithm.deletePolicy(AttributeSubject, AttributeObject) (abac_blockchain.abac)
      ↳ Blockchain.deleteBlock(AttributeSubject, AttributeObject) (abac_blockchain.blockchain)
        ↳ SHA256 (abac_blockchain.encryption)
        ↳ SHA256.encrypt(Any?) (abac_blockchain.encryption)
        ↳ AttributeSubject.getUserId() (abac_blockchain.attributes)
        ↳ AttributeSubject.getRole() (abac_blockchain.attributes)
        ↳ AttributeSubject.getGroup() (abac_blockchain.attributes)
        ↳ AttributeObject.getDeviceld() (abac_blockchain.attributes)
        ↳ DataBase.deleteRecord(String) (abac_blockchain.database)
        ↳ DataBase.countRows() (3 usages) (abac_blockchain.database)
        ↳ DataBase.updatePreviousHash(String, String)(3 usages) (abac_blockchain.database)
        ↳ DataBase.getPreviousHash(Int)(2 usages) (abac_blockchain.database)
        ↳ DataBase.getHashValue(Int) (abac_blockchain.database)
  
```

Fig 5: Execution calls

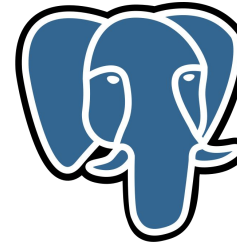


## Working Environment



**IntelliJ IDEA**

IntelliJ IDEA provides a set of inspections that are built-in static code analysis tools. They help you find potential bugs, locate dead code, detect performance issues, and improve the overall code structure



**PostgreSQL**

PostgreSQL is a powerful, open source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads.

Fig 6,7[6][7]: IntelliJ IDEA and PostgreSQL Icon

# Working Environment Snapshots

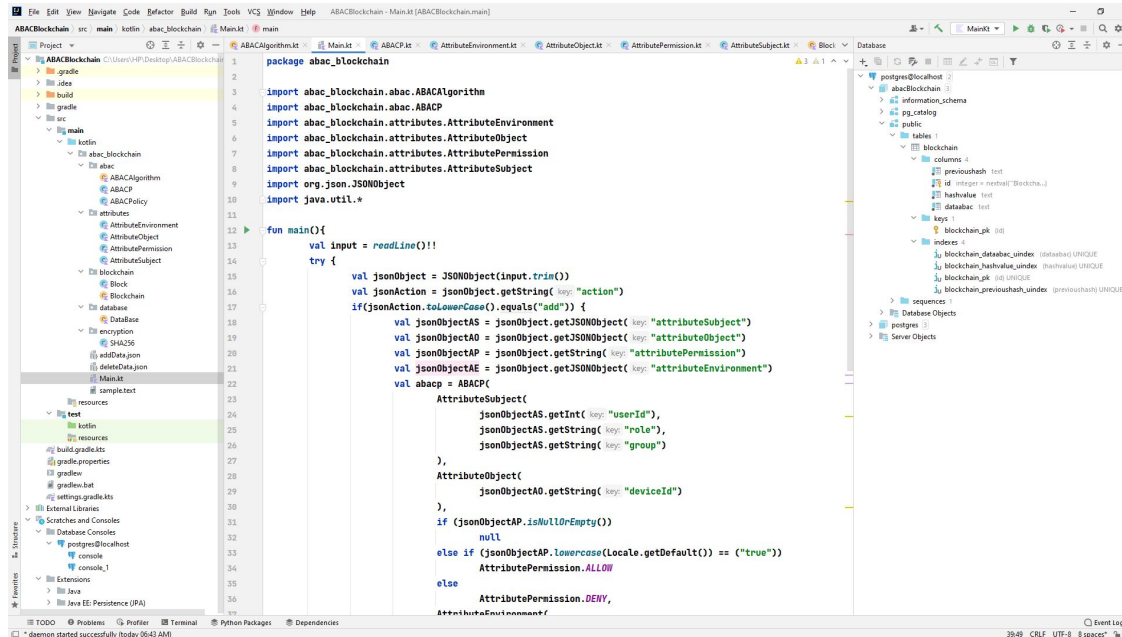


Fig 8: Working environment snapshots

## Testing:-

Input	Description	Result
<pre>{   "action": "add",   "attributeSubject": {     "userId": 58102,     "role": "sensor",     "group": "detection"   },   "attributeObject": {     "deviceId": "AX0101"   },   "attributePermission": "true",   "attributeEnvironment": {     "createTime": 1000,     "endTime": 10000   } }</pre>	Data provided to add block action is accurate	Successful
<pre>{   "action": "delete",   "attributeSubject": {     "userId": 58473,     "role": "sensor",     "group": "detection"   },   "attributeObject": {     "deviceId": "AX0123"   } }</pre>	Data provided to delete block action is accurate	Successful

**Table 5 :Test case table**



## Testing:-

Input	Description	Result
<pre>{   "action": "add",   "attributeSubject": {     "userId": 58102,     "group": "detection"   },   "attributeObject": {     "deviceId": "AX0101"   },   "attributeEnvironment": {     "createTime": 1000,     "endTime": 10000   } }</pre>	Data provided to add block action is not accurate, here missing of data or inaccurate data is given	Unsuccessful
<pre>{   "action": "delete",   "attributeSubject": {     "userId": 58473   },   "attributeObject": {     "deviceId": "AX0123"   } }</pre>	Data provided to delete block action is not accurate, here missing of data or inaccurate data is given	Unsuccessful

**Table 6 :Test case table**

## Experimental Results:-



```
Run: MainKt
"C:\Program Files\Java\jdk-15.0.2\bin\java.exe" ...
{"action": "add", "attributeSubject": {"userId": 58102, "role": "sensor", "group": "detection"}, "attributeObject": {"deviceId": "AX0101"}, "attributePermission": "true", "attributeEnvironment": {"creat
Process finished with exit code 0
```

The screenshot shows an IDE's Run console. The top line indicates the command executed: `"C:\Program Files\Java\jdk-15.0.2\bin\java.exe" ...`. The second line shows a JSON object being passed as an argument: `{"action": "add", "attributeSubject": {"userId": 58102, "role": "sensor", "group": "detection"}, "attributeObject": {"deviceId": "AX0101"}, "attributePermission": "true", "attributeEnvironment": {"creat`. The third line shows the result: `Process finished with exit code 0`. The IDE interface includes a toolbar on the left with icons for Run, Stop, Debug, and other actions, and a bottom status bar with tabs for Run, TODO, Problems, Profiler, Terminal, Python Packages, Build, and Dependencies.

Fig 9: input add block data in JSON Format

## Experimental Results:-



```
Run: MainKt
"C:\Program Files\Java\jdk-15.0.2\bin\java.exe" ...
{"action": "delete", "attributeSubject": {"userId": 58473, "role": "sensor", "group": "detection"}, "attributeObject": {"deviceId": "AX0123"}}
Process finished with exit code 0
```

Fig 10 : input delete block data in JSON Format

## Experimental Results:-

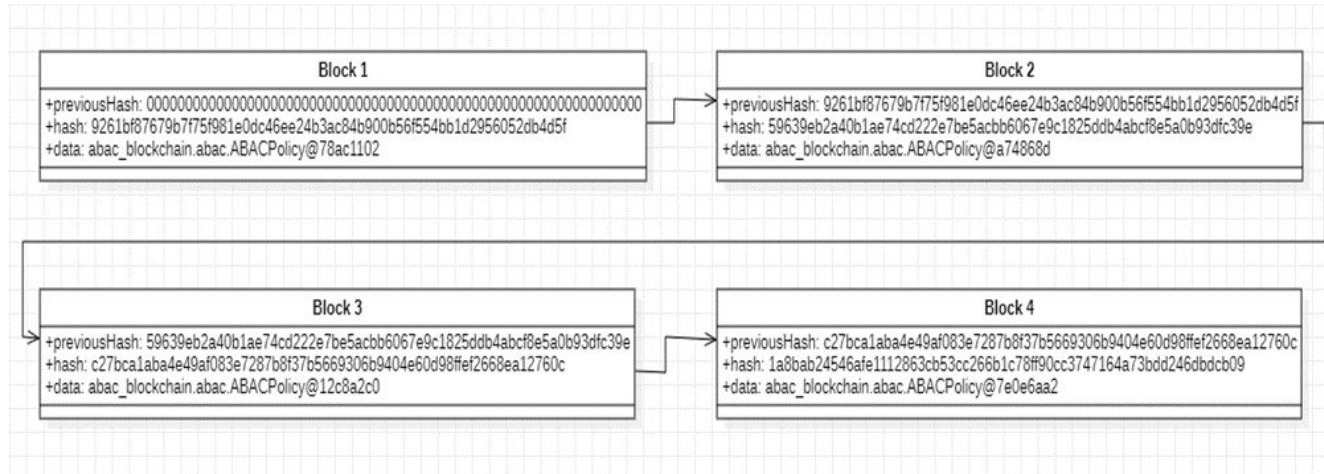


Fig 11 : Blocks in blockchain

[illegible]

**Fig 12 : Block data in database**



## Conclusion

- This work proposed a Blockchain Attribute Based Access Control model (ABAC) that employs the blockchain as the trusted center of the access control model and implements the access control policy.
- The data stored on the blockchain is trustworthy and credible because of its tamper-proof and no single point of failure features.
- The proposed model is fully decentralized (no third-party required), user-friendly, user-transparent, fault-tolerant, scalable, and compatible with a wide range of access control models.



## Future Work :-

- In the future, we will work on the security and privacy of IoT data from unauthenticated edge nodes.
- We are planning to test the security of the system against access control attacks such as forgery attacks, injection attacks, and man-in-the-middle attacks.
- For a better test of the proposed model, we are planning to use IoT physical devices to test the reliability and throughput of the system.
- Finally, physical infrastructure can have an impact on blockchain performance..



## References

- [1] <https://www.foley.com/en/insights/publications/2021/08/types-of-blockchain-public-private-between>
- [2] W. Liang, M. Tang, J. Long, X. Peng, J. Xu, and K.-C. Li, “A secure fabric blockchain-based data transmission technique for industrial Internet-of-Things,” *IEEE Trans. Ind. Inf.*, vol. 15, no. 6, pp. 3582–3592, Jun. 2019
- [3] O. Novo, “Blockchain meets IoT: An architecture for scalable access management in IoT,” *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1184–1195, Apr. 2018.
- [4] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, “Smart contract based access control for the Internet of Things,” *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1594–1605, Apr. 2019.
- [5] G. Papadodimas, G. Palaiokrasas, A. Litke, and T. Varvarigou, “Implementation of smart contracts for blockchain based IoT applications,” in *Proc. 9th Int. Conf. Netw. Future (NOF)*, Nov. 2018, pp. 60–67.
- [6] <https://en.wikipedia.org/wiki/PostgreSQL>
- [7] [https://en.wikipedia.org/wiki/IntelliJ\\_IDEA](https://en.wikipedia.org/wiki/IntelliJ_IDEA)





# THANK YOU