Spring 2019

Combining Blockchain and Swarm Robotics to Deploy Surveillance Missions

Ardalan Razavi
San Jose State University

COMBINING BLOCKCHAIN AND SWARM ROBOTICS TO DEPLOY SURVEILLANCE MISSIONS

A Thesis

Presented to

The Faculty of the Department of Computer Engineering

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Ardalan Razavi

May 2019

The Designated Thesis Committee Approves the Thesis Titled

COMBINING BLOCKCHAIN AND SWARM ROBOTICS TO DEPLOY SURVEILLANCE MISSIONS

by

Ardalan Razavi

APPROVED FOR THE DEPARTMENT OF COMPUTER ENGINEERING

SAN JOSÉ STATE UNIVERSITY

May 2019

David C. Anastasiu, Ph.D.	Department of Computer Engineering
Gheorghi Guzun, Ph.D.	Department of Computer Engineering
Gokay Saldamli, Ph.D.	Department of Computer Engineering

ABSTRACT

COMBINING BLOCKCHAIN AND SWARM ROBOTICS TO DEPLOY SURVEILLANCE MISSIONS

by Ardalan Razavi

Current swarm robotics systems are not utilized as frequently in surveillance missions due to the limitations of the existing distributed systems' designs. The main limitation of swarm robotics is the absence of a framework for robots to be self-governing, secure, and scalable. As of today, a swarm of robots is not able to communicate and perform tasks in transparent and autonomous ways. Many believe blockchain is the imminent future of distributed autonomous systems. A blockchain is a system of computers that stores and distributes data among all participants. Every single participant is a validator and protector of the data in the blockchain system. The data cannot be modified since all participants are storing and watching the same records. In this thesis, we will focus on blockchain applications in swarm robotics using Ethereum smart contracts because blockchain can make a swarm globally connected and secure. A decentralized application (DApp) is used to deploy surveillance missions. After mission deployment, the swarm uses blockchain to communicate and make decisions on appropriate tasks within Ethereum private networks. We set a test swarm robotics system and evaluate the blockchain for its performance, scalability, recoverability, and responsiveness. We conclude that, although blockchain enables a swarm to be globally connected and secure, there are performance limitations that can become a critical issue.

ACKNOWLEDGMENTS

I want to thank my friend and advisor Professor Gokay Saldamli, Ph.D., who advised and supported me on this work. Nevertheless, I would like to thank my family and friends for supporting and expecting nothing less than excellence from me

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CMPE	Department of Computer Engineering
DApp	Decentralized Application
EVM	Ethereum Virtual Machine
IoT	Internet of Things
IPFS	InterPlanetary File System
PoW	Proof of Work
PoS	Proof of Stake
SJSU	San José State University

1 INTRODUCTION

Swarm robotics is a field that studies how a large number of simple robots manage and coordinate complex tasks as a group using local knowledge. Local knowledge can be defined as the information gathered by each robot's sensors. Swarm robotics systems have been used in various industries to perform specific tasks without global knowledge or any decision making as a group [1]. For instance, the automotive industry has been using robots that perform a task together without global knowledge. Swarms of this kind have a high dependency on each other. As Navarro and Matia [2] mention in their paper, an ideal swarm should have the following advantages:

- Improved performance by using parallel computing
- Higher achievement by using the swarm's processing power
- Distributed environment sensing
- Higher fault tolerance

As mentioned in [3], swarm robotics systems are facing problems due to the lack of global awareness. Also, Ferrer mentions [4] "One of the main axioms in the swarm robotics field has been the absence of global knowledge or explicit communication models between swarm robots." This problem prevents swarms from behaving dynamically, and it creates deadlocks while performing tasks. The decisions are made based on local knowledge. As a result, the traditional swarm robotics systems have limitations when it comes to jobs that require cooperation and consensus. Blockchain is a technology that has shown great potential for managing distributed systems. After observing Bitcoin's use of blockchain, new projects like Ethereum were developed. Ethereum introduced the idea of smart contracts which make it possible to store logic (code) on the blockchain. The combination of blockchain and swarm robotics can bring security and global awareness to a distributed swarm of robots. In this paper, we

simulate a swarm robotics system that uses the Ethereum blockchain to determine its usability in the swarm robotics field.

2 BACKGROUND

2.1 Blockchain

Blockchain is an immutable digital ledger system that is distributed among computers in a network. Blockchain chains the blocks of data to each other using a cryptographic hash without a central authority. The ledger grows in size with time [5]. The data in a ledger are immutable after blocks are verified. Immutabilitable data brings security and transparency. Fig. 1 shows a simple overview of how blockchain technology works. Three concepts should be distinguished when talking about

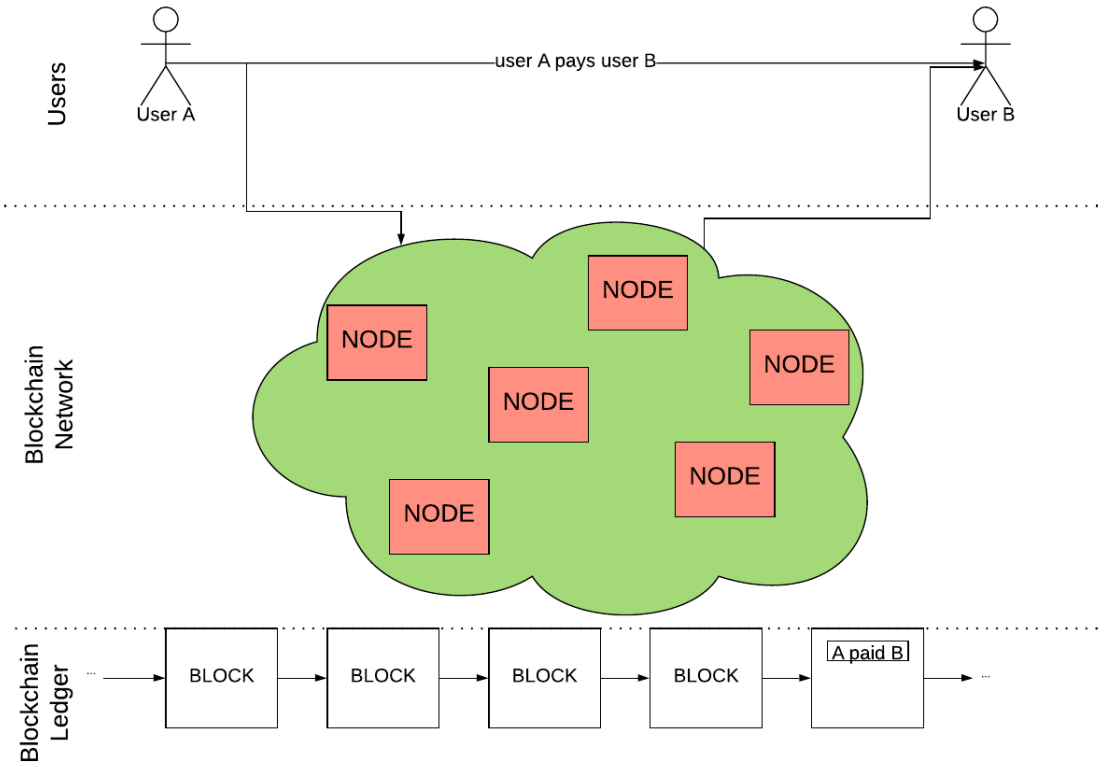

Fig. 1. Blockchain overview.

blockchain; The blockchain, the protocol, and the cryptocurrency [6]. Cryptocurrencies are digital money stored on a blockchain network. Protocols are the rules within the

blockchain network. These components differ from a blockchain to another. The first usage of blockchain technology was in the form of cryptocurrency (Bitcoin). Each cryptocurrency owner can make electronic transactions from his or her account to another. Each electronic transaction is signed by the account owner's private key and verified by network participants. Each block consists of a header and a list of transactions which are chosen from the participant's transaction pool. The participants' job is to verify all of the transactions in a block and solve a puzzle unique to that block. Once the puzzle is solved, the answer is added to that block header. Then, the block is broadcasted to the network where all of the other participants verify the validity of the block. These participants are rewarded with cryptocurrency.

2.1.1 Hashing

An important component of the blockchain is the use of cryptographic hash functions. Hashing is a method of calculating a unique fixed-size output from an input. Any change to the input will change the hash value. As shown in Table 1, the slightest change to the input alters the output of the SHA256 function. The hash function algorithms are one-way. One-way means that the input cannot be calculated from its output. It is not feasible to compute the same output from different inputs. As a result, hash algorithms are collision resistant. Secure Hash Algorithm (SHA) is a popular choice among the blockchain technologies. The SHA algorithm is supported by most computer hardware. One of the usages of this type of algorithm in blockchain is hashing each block's transaction list and creating a fingerprint. If a single transaction changes then the fingerprint is different. This way any minor change within a block is easy to detect.

Table 1

SHA256 Hash Function Input and Output

Input	Output
blockchain	EF7797E13D3A75526946A3BCF00DAEC9F C9C9C4D51DDC7CC5DF888F74DD434D1
Blockchain	625DA44E4EAF58D61CF048D168AA6F5E 492DEA166D8BB54EC06C30DE07DB57E1
Blockchains	E7BFC01C0D3B22212BFC777460E7B3B9 F3FB03E512F70FD4BD43AC669348BDE3

2.1.2 Asymmetric Key Cryptography

A fundamental technology used by blockchain technologies is asymmetric key cryptography [7]. This cryptographic method uses two keys to encrypt and decrypt a message. A public key may be made public for anyone who wants to send data to the person holding the corresponding private key. The private key is kept as a secret on the contrary to the public key. A public and a private key are generated mathematically. These two keys are related to each other. Data encrypted with a public key can be decrypted with the related private key. Also, data encrypted with a private key can be only decrypted with the related public key. The asymmetric key cryptography is utilized in blockchain systems for the following use cases:

- Private keys sign transactions
- Public keys create accounts
- Public keys verify signatures

2.1.3 Consensus Algorithms

A consensus is a task of reaching an agreement on data within distributed processes. In a distributed, open, and trustless system nodes can act maliciously. Blockchain technology uses different consensus algorithms to verify state changes validity within a blockchain network. All of the participant nodes in a network work together to verify the system integrity using a consensus algorithm. This eliminates a need for a

centralized authority to verify the state changes. It also empowers all of the participants in a blockchain network to vote on state changes. Two of the most popular consensus algorithms are Proof of Work (PoW) and Proof of Stake (PoS).

2.1.3.1 PoW: In PoW, all participant nodes compete to solve the computationally expensive puzzle for each block. A node can suggest a new block to the blockchain network once the answer to the puzzle is found. A node is called miner if it engages in PoW consensus by solving these puzzles. Solving puzzles are challenging, but verifying the solutions are quick and easy. This way, all of the participant nodes in a network can quickly verify the validity of the proposed solutions. A block is added to the blockchain if other nodes confirm a solution. In PoW, miners compete against each other to solve puzzles and add blocks to a blockchain. A hash function is used by a miner to find a 32-bit unique number for each block. This unique number is called nonce. A miner adds a generated nonce to the block header and hashes the block header using a hash function. The result is a 256-bit hash value that should start with a certain number of zeros. Finding a nonce is a brute force method. A miner might find a valid nonce in a first try or it might take a long time.

2.1.3.2 PoS: PoS also ensures that the miners agree on the same branch of data in a blockchain network. PoS eliminates the race condition produced in PoW. In PoS, a miner is elected with a probability which is proportional to the amount of stake it owns in the blockchain network. This way, miners are not competing to solve a mathematical problem. The miner stake is at risk if a false block is committed to the blockchain network. A false block is a block in which one or multiple transactions are not valid. The blockchain keeps track of network validators. A validator can be anyone who holds a blockchain's cryptocurrency. A validator needs to send a special type of transaction that locks up the stake into a deposit transaction. A new validator is selected during each

time slot and can create a new block that points to a previous block. PoS reduces electricity consumption and centralization risks in a blockchain network.

2.1.4 Smart Contracts

A smart contract is a self-executing code that is stored on a blockchain. The miners are responsible for verifying each smart contract. A smart contract can take actions at specified times or based on the occurrence of actions or events defined in its code. A smart contract is not necessarily smart. The smartness of a smart contract comes from the data and the information that it receives and the code that directs it. An example of a smart contract is decentralized voting. Voters can vote using methods defined in the smart contract. The participant nodes in the network verify the validity of votes. A smart contract can do what its code instructs it to do based on the input. The miners verify the smart contract transactions. Each instruction in a smart contract has a cost associated with it. As a result, a user is responsible to pay fees when sending transactions to a smart contract. These fees encourage mining nodes to participate in a network, and they prevent malicious users from performing a denial of service by executing a computation-heavy smart contract function.

2.1.5 Projects

2.1.5.1 Bitcoin: Bitcoin is the first cryptocurrency build on top of a blockchain. Bitcoin is created by Satoshi Nakamoto. Satoshi's motive was to improve the banking and financial industry by making it decentralized. Bitcoin is used as a digital currency. Bitcoin transactions are stored on distributed ledgers. Bitcoin operates using nodes that are connected using peer-to-peer connections. As bitcoin founder Nakamoto mentions in Bitcoin's white paper, "The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work" [8]. Bitcoin uses the PoW consensus algorithm to verify the blocks. Bitcoin nodes solve the puzzle for each block and chain those blocks

together by linking the previous block to the newly created one. The PoW is considered inefficient and not substantial because of race condition between all of the nodes to solve a block. All of the nodes in a network are trying to verify the same block in parallel until one node finds the solution and broadcast it to the network.

2.1.5.2 Ethereum: Ethereum blockchain is a successor of Bitcoin. As of today, the Ethereum blockchain uses the PoW consensus algorithm. As Ethereum founder, Buterin, mentions in his Ethereum white paper [9], Ethereum is a blockchain with a built-in Turing-complete programming language which allows anyone to write code that gets stored on the Ethereum blockchain. Decentralized applications (DApp) can create their own arbitrary rules for ownership, transaction formats, and state transition functions within the smart contracts. DApps are applications that interact with the Ethereum blockchain network instead of centralized servers. In the Ethereum blockchain, a smart contract gets executed in the Ethereum Virtual Machine (EVM), and its code is written in Solidity (Ethereum smart contract language). Each interaction with the Ethereum blockchain is either a call or a transaction. A call doesn't change the blockchain state. On the other hand, a transaction changes the state of the blockchain. Blockchain browser bridges are required by DApps. Metamask is a browser bridge that lets a user interact with a DApp in a browser. Ethereum nodes are the gateways to any Ethereum network. A machine can become a node in an Ethereum network by running an Ethereum client. There are different Ethereum clients developed. The most famous ones are Geth, Parity, and eth (cpp-ethereum).

2.2 Swarm Robotics

Swarm Robotics is inspired by self-organizing behaviors of insects in nature. Swarm robotics is the study of how a group of simple agents can be designed such that a global state can be reached by agents' communication and observation of an environment [10]. For instance, ants work in swarms to accomplish tasks which can not be accomplished

by a single ant. Swarm robotics is a newly emerging field in engineering which aims at modeling a system which consists of simple agents with micro-tasks assigned to them. All agents in these type of systems perform their micro-tasks that create a well-organized system. These systems can accomplish tasks that are beyond the power and capabilities of a single agent. The study of the insects in nature has shown that communication and data exchange between all of the insects in the swarm is not centralized [11]. By observing the swarm of insects in nature, researchers learned the following characteristics of swarms:

- Robustness: The Swarm should be able to operate in case of disasters and failures. For instance in a ants swarm, if an ant fails, another one can accomplish the task without any disruption in the overall swarm. There are a few factors which make a swarm robust:
  - Redundancy: An individual failure can be compensated by other individuals.
  - Decentralized: There is no single point of failure in the swarm. If one part of the swarm is damaged, then the swarm can continue on its task with effective communication.
- Flexibility: One of the main characteristics of the swarm is to be able to be flexible and respond to different situations in a suited and logical way. For instance, ants act differently when pulling or lifting something together.
- Scalability: A swarm should be able to operate in the same way with a change of workload.

3 RELATED WORK

3.1 Smart Cities

Cisco is currently working on making cities smart around the globe by using Internet of Things (IoT) devices. The sensors around the city gather information and send them to a centralized control center which has the logic layer built in. All type of useful information can be collected using different sensors. For example, passive infrared sensors can be used to track the status of all parking spots around a city. None of the existing systems currently are capable of communicating and publishing their data without the need for a central control system. Users need to access the central server if they want to query the data from all of the sensors. According to IEEE, two of the biggest challenges of large IoT projects like smart cities are security and connectivity [12]. By having so many sensors installed in cities, a centralized server and client architecture is not scalable. An ideal design is to have peer-to-peer communications between all participating devices within an IoT network.

3.2 Blockchain and Supply Chain

A supply chain can be defined as a set of entities directly involved in flows of products, finance, services, and information from a source to customers [13]. One of the biggest concerns in the supply chain is how to track all of the activities and actions from the starting point, logistics, and eventually delivery. Many companies started looking into blockchain technology to bring transparency to the supply chain. According to IBM [14], using blockchain in the supply chain will help with the following issues:

- Reducing or eliminating fraud and errors since the transactions are immutable
- Reducing the time spent on documentation by humans
- Bringing transparency in all operations
- Increasing trust between parties

The Blockchain in Trucking Alliance (BiTa) was formed by some of the technology giant (for instance UPS and FedEx) executives to create a forum for the development of the blockchain standards used in the supply chain. In addition, companies like VeChain are working to build a global enterprise level public blockchain platform for supply chains. VeChain's goal is to connect blockchain technology to the real world by providing a comprehensive governance structure. VeChain wants to accomplish these tasks by the integration of IoT devices with real-world applications. VeChain makes it possible for products' end-users and suppliers to track products' journey from start point to end point. This provides transparency and security for the supply chain processes.

3.3 Golem

Not every individual has access to a supercomputer to solve complex problems. Project Golem is trying to solve this issue by creating a distributed and decentralized supercomputer available to all of the participants in the Golem blockchain network. It enables people to access the decentralized computing power of other people's computers. The Golem blockchain relies on people running Golem nodes. Golem allows individuals to earn money from unused computation resources that they own. Golem is built on top of the Ethereum blockchain and connects computers in a peer-to-peer network. Users can upload computing jobs to the Golem network, and the job will be distributed between a network of peer-to-peer connected computers. Computers (owners) that complete the jobs are called "providers". Users who request computing jobs are called "requesters". The value (token) stored on Golem network is called GNT. GNT is the form of incentive transferred from requesters to providers. Each provider earns GNT based on the amount of computing power used on a job. Golem uses Ethereum smart contracts to manage each job's payments. Payments are distributed among the providers without a need for a centralized authority. As shown in Fig. 2 providers register their hardware (computing) to the Golem network. In addition,

developers create and build software solutions that can be utilized by users. Requesters' demand is fulfilled by providers' registered hardware. Both developers and providers are rewarded for their contributions to the Golem network [15].

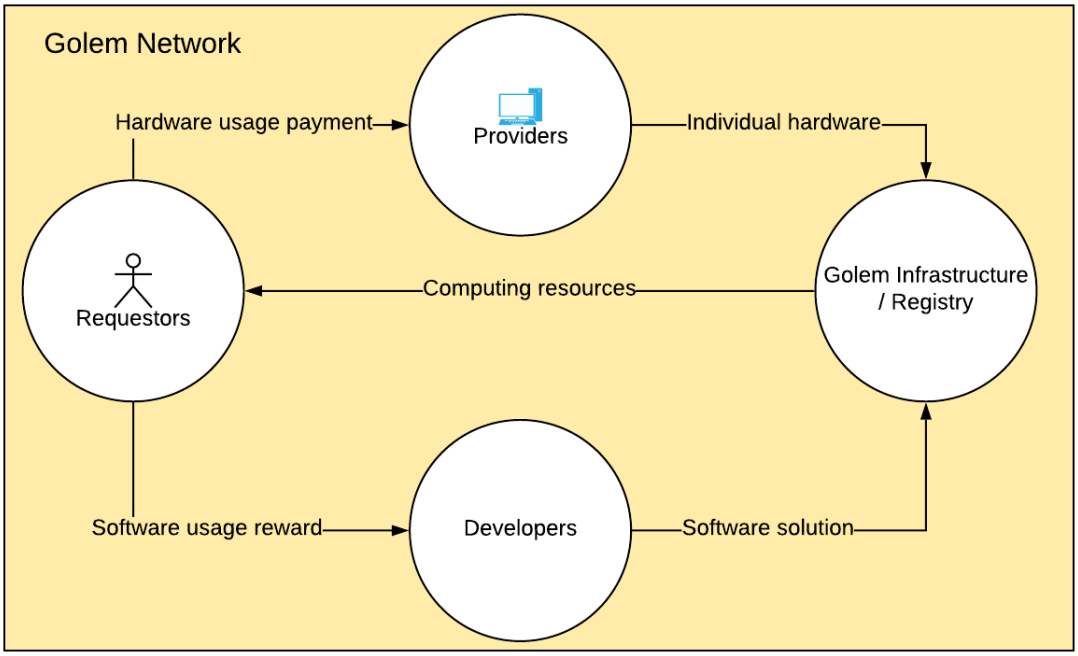

Fig. 2. Overview of Golem's network infrastructure.

3.4 EtherDelta

Etherdelta is a decentralized cryptocurrency trading platform and a DApp. Users can exchange Ethereum based tokens directly with other users. Users need to use browser bridges like Metamask or their Ethereum token wallet private key to interact with Etherdelta. Etherdelta operates using smart contracts that are stored on Ethereum blockchain. These smart contracts provide exchange functionalities.

3.5 Swarm-Bot

Swarm-Bot is the first Search and Rescue (SAR) project in swarm robotics. The project was inspired by colonies of ants. The project replicated ants' activities like

transporting objects, building nests, and building bridges to move from one point to another. Each s-bot was designed as a simple but fully autonomous unit capable of moving, sensing the environment, and acting based on swarm consensus [16]. In addition, each robot was equipped with sensors and communication devices to communicate with other robots. The swarm acted only based on local knowledge.

4 SYSTEM DESIGN

The designed swarm uses Ethereum private blockchains and smart contracts to communicate and store data. The control center is a DApp that uses smart contracts to interact with the Ethereum private blockchain. The control center deploys a surveillance mission contract per user request. In addition, after deployment, the user can monitor the deployed swarm's activities by interacting with the Ethereum private blockchain. Each robot is deployed with a code that provides the capabilities for the robot to interact with the blockchain and perform its tasks.

4.1 Blockchain

One of the main characteristics of the blockchain is decentralization. Fig. 3 shows traditional centralized network on the left and the decentralized on the right. As mentioned in the Ethereum white paper [9] "Decentralized consensus gives Ethereum extreme levels of fault tolerance, ensures zero downtime, and makes data stored on the blockchain forever unchangeable and censorship-resistant.". The blockchain is responsible for validation and storing all of the actions performed in the swarm by all participants. By using blockchain in the swarm, the problem of having a point of failure is eliminated.

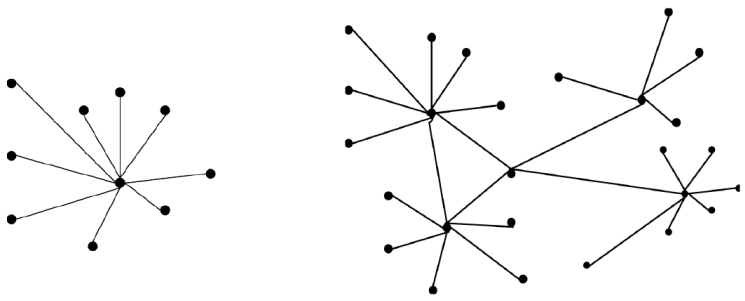

Fig. 3. Centralized vs decentralized network.

4.1.1 Ethereum Private Blockchain

The swarm communication, global knowledge, and command infrastructure are built on top of an Ethereum private blockchain. Robots use the PoW consensus and smart contracts to form a swarm.

4.1.1.1 The main private blockchain: One blockchain is deployed only for the sole purpose of the control center smart contract deployment. This private blockchain is called the main private blockchain network.

4.1.1.2 The dedicated private blockchain: Another type of private blockchain is deployed for each mission. This blockchain network is a dedicated private blockchain network. A dedicated private blockchain starts with a mission request and ends once the mission terminates.

4.1.2 Smart Contracts

There are two types of smart contracts deployed on the system blockchain networks. First, the control center smart contract is deployed on the main private blockchain network. Second, as many as needed surveillance mission smart contracts are deployed on a dedicated private blockchain.

4.1.2.1 Control Center Smart Contract: The control center smart contract is the first transaction committed to the main private blockchain network. The contract is the main component of the control center DApp. All users' and robots' registration processes use this smart contract. After registration, participants are able to interact with the main private network. More information is provided in 4.7.1 about the registration flow.

4.1.2.2 Rescue Mission Smart Contract: A user uses the control center DApp to create and deploy a surveillance mission smart contract. Employed robots' information needs to be passed into the contract's constructor. Each contract is constructed using each robot's address and initial location. No other robot can interact

with the smart contract after deployment. The mission robots update their information using this contract. Each robot's speed, movement, and direction are stored on the contract. More information is provided in 4.7.2.

4.1.3 Node types

The Ethereum network has full and light nodes. Fig. 4 shows the relationship between the light and full nodes in an Ethereum network. In each swarm, light and full nodes are used to reduce the storage size and computation time in a swarm.

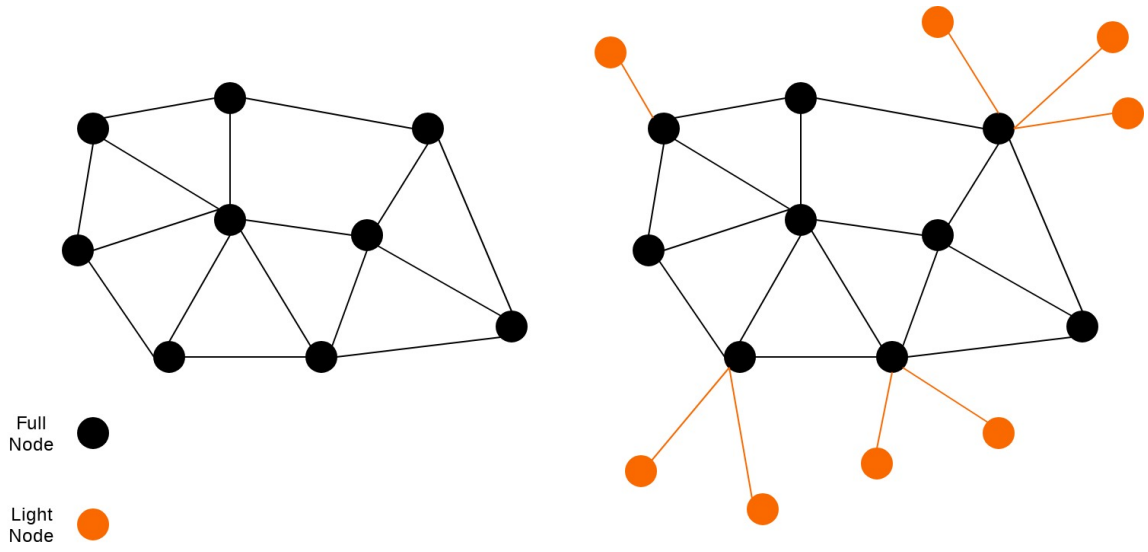

Fig. 4. The blockchain nodes.

4.1.3.1 Light Node: A light node as the name implies does not store the blockchain history locally; it gets the current state of the blockchain. In addition, a light node does not verify transactions that are committed to a blockchain. As a result, a light node is a great candidate for the robots that have computation power and storage limitations.

4.1.3.2 Full Node: A full node is responsible for storing the full history of a blockchain. In addition, a full node performs all of the transactions' verifications (mining). In a network, light nodes rely on full nodes to perform their tasks. A full node

stores all the records from the first block (genesis block) to the latest block of a blockchain. A swarm's full nodes execute all of the transactions committed to the network.

4.1.3.3 Bootnode: In each swarm, one bootnode is deployed. The only task of a bootnode is to keep track of all of the nodes in a network. A bootnode only promotes awareness to the nodes about the existence of other nodes in the network. It does not perform any blockchain related tasks in the swarm. The nodes in a network use a bootnode to find and connect to other peers.

4.2 Robots

Robots are the service providers in the system. Each robot must have a daemon process running at all times on its system. The daemon code is downloaded from the control center DApp. The daemon provides the following capabilities:

- Communication with the main private blockchain
- Communication with the dedicated private blockchain
- Deployment of a dedicated private blockchain node by running a Geth client
- Communication with IPFS network

Each robot needs to be registered in the system by using the control center smart contract. Only a registered robot can interact with the system. A registered robot is activated when a user requests a surveillance mission from the control center. Before the activation, a robot constantly checks to see if it is activated. Once activated, the robot becomes a full or a light node by running a Geth client. The robot uses a Geth client to connect to other activated robots. Once a robot joins a swarm, it constantly updates the following information:

- Location
- Speed
- Sensors data

- Upload vision data if needed

4.3 The Control Center DApp

The control center DApp is deployed on the main private Ethereum blockchain. The main component of the control center is the control center smart contract. Fig. 5 shows the deployment process of the contract.

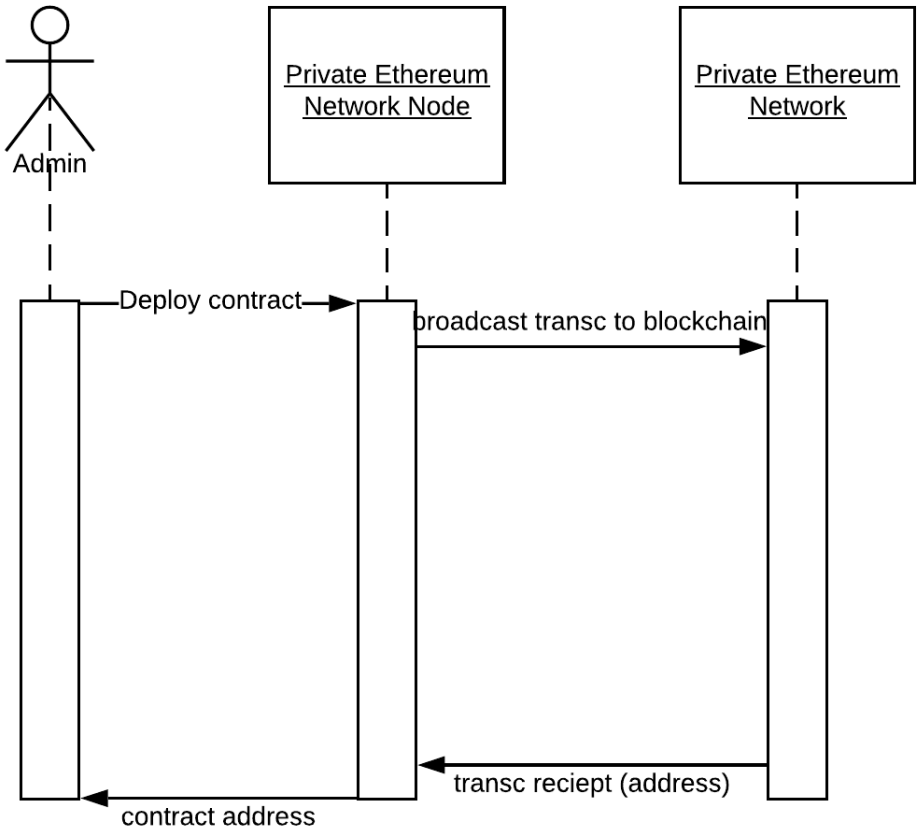

Fig. 5. The control center smart contract deployment.

First, the admin compiles the contract using solidity compiler to get the bytecode and contract's interface. Then, a transaction is sent to a private main network's node by including the bytecode. The node broadcasts the transaction to the network. The

transaction is confirmed with the PoW mining process. Next, the contract's address is retrieved from the transaction's receipt. The control center DApp then stores the contract's address and interface for future uses. A user can use the control center DApp to create and deploy a surveillance mission contract. Below is the overview of the steps performed by a user and the robots before and after each surveillance mission deployment:

- Robots:
  - Register using the control center smart contract
  - Download the surveillance mission code
  - Check the control center smart contract to see if they have been activated
  - Join the dedicated mission blockchain network
- Users:
  - Registers using the control center (DApp)
  - Provides the following data for the surveillance mission:
    - The number of robots needed
    - Approximate Location (longitude and latitude) of the job
  - Monitor the swarm after activation using the control center DApp

The control center DApp provides monitoring interfaces for different private blockchains. Each working swarm periodically stores its recent status in the blockchain. The DApp provides two monitoring interfaces. One monitoring interface is dedicated to a swarm's performance. The monitoring interfaces use an open-source monitoring tool called "eth-netstat". Second monitoring interface is dedicated to the swarm's dedicated private network status. Location, speed, number of active agents, and swarm data are listed for a user to monitor the status of the working swarm.

4.4 Storage Methods

InterPlanetary File System (IPFS) and the blockchain smart contracts are the two methods of storage used in a swarm. Fig. 6 shows an overview of smart contract and IPFS relation in a swarm.

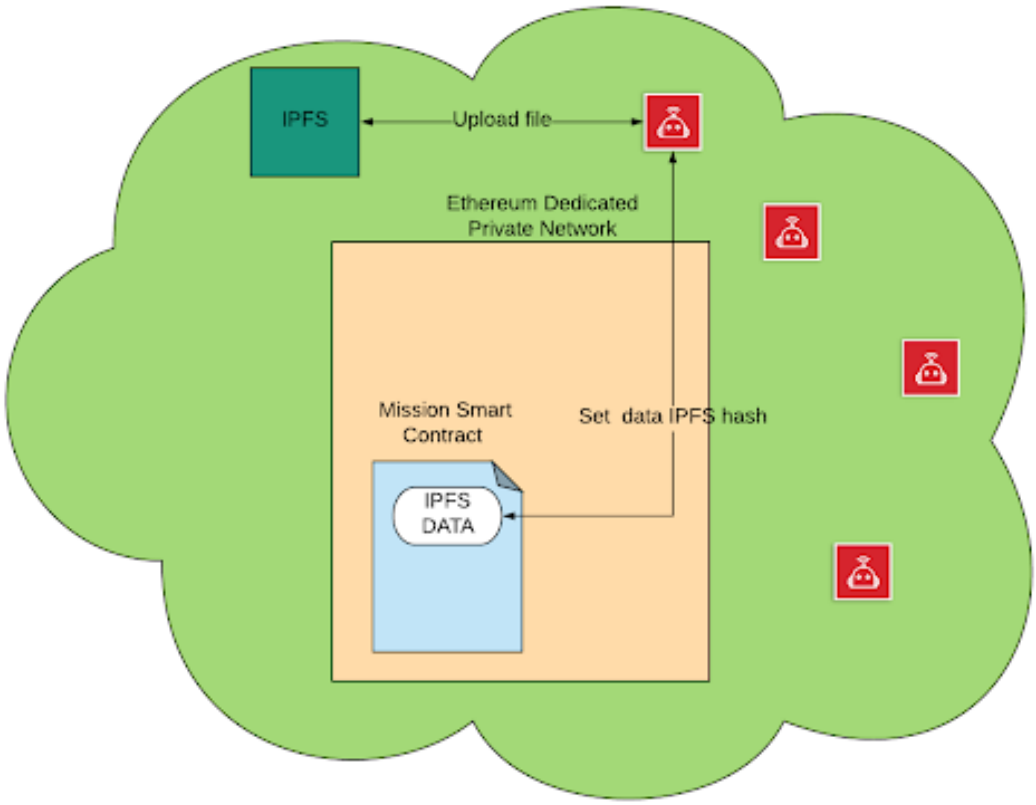

Fig. 6. Swarm storage.

Each robot is capable of uploading camera images to the IPFS network at the time of surveillance. Images are uploaded to the IPFS network by a robot. Then, a hash is returned from an IPFS node. Next, the hash is stored on the mission smart contract by the robot. A user can view the image by using the control center DApp monitoring interface. The control center DApp gets the hash by calling and retrieving the image

address (IPFS hash) from the mission smart contract. Then, it downloads the image from the IPFS network and shows it to a user.

4.4.1 InterPlanetary File System (IPFS)

As Benet mentions in his paper [17] “The InterPlanetary File System (IPFS) is a Peer-to-Peer distributed file system that seeks to connect all computing devices with the same system of files.”. Robots throughout the life span of a swarm use IPFS to upload and retrieve files from the distributed and decentralized file system. The command center is able to pull the data from the IPFS network without putting any load on robots. Files stored on the IPFS network can not be modified and tempered due to the usage of cryptographic hash for each file. The address of each file is the hash of its content. Fig. 7 shows the difference between traditional and IPFS storage design. In a centralized file system, there is a high dependency on a single point of failure. On the other hand, all nodes are connected without a single point of failure in the IPFS network. IPFS uses content-based addressing instead of location-based addressing. Once a file is stored on IPFS, its content hash is returned. The hash can be used to retrieve the file from the IPFS network.

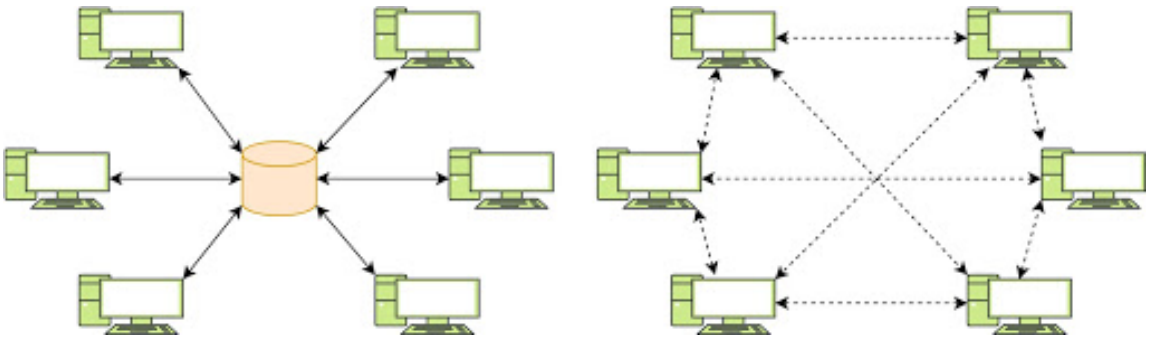

Fig. 7. Traditional file system vs InterPlanetary File System.

4.4.2 Smart Contracts

Smart contracts are used as a storage utility in a swarm. Users, robots, and surveillance missions information is stored in the smart contracts. Users and robots registration data are stored in the control center contract. In addition, robots store their current status in a surveillance mission contract. Storing large data in smart contracts are expensive because all nodes need to store the blockchain ledger locally. As a result, using IPFS and smart contracts together is the best solution for storing large data in a swarm. First, data is stored on the IPFS network. Then, the returned hash from the IPFS network is stored in a smart contract.

4.5 Communication Methods

Swarm participants use Peer-to-Peer connections to interact with other blockchain participants and sync data. Robots use the Ethereum Geth client to connect to the blockchain. Geth is a go language Ethereum client that is capable of interaction with the Ethereum network. For data transfer, robots use a protocol named RLPx. RLPx enables robots to transfer encrypted and serialized data. Robots agree on a cryptographic protocol for the encryption and signing of the data. These protocols are “eth” for full nodes and “les” for light Ethereum nodes. After the protocol exchange, subsequent messages are transmitted based on the chosen protocol.

4.6 Security

Asymmetric key cryptography is the fundamental technology used by blockchain. Asymmetric key cryptography uses private and public keys to decrypt and encrypt data. In a swarm, robots public key is used to create their blockchain addresses. Robots private keys are stored locally in each robot’s storage. The Asymmetric key cryptography utilized by blockchain has the following uses [18]:

- A public key is used to create blockchain addresses.

- A public key is used for signature verifications (signed by private keys).
- A private key is used for the digital signature of each transaction.
- Keys used for the resource (smart contract) action restriction.

Furthermore, it is possible to use a multiple signature (multisig) method for securing a critical action defined in the smart contract. The multisig method requires more than one robot to approve the action that a single robot wants to perform. Fig. 8 shows the example of usage of a multisig transaction within a swarm. Robot E wants to relocate and move from location A to location B. As a result, robot E adds its action to the required permission action list of the smart contract. Robots A, B, and C are authorized to participate in the consensus to vote for robot E relocation. Robot B and C vote yes to the relocation of Robot E. The transactions are signed by their private keys and verified in the smart contract. Robot E relocates due to the majority of voters agree on the relocation action.

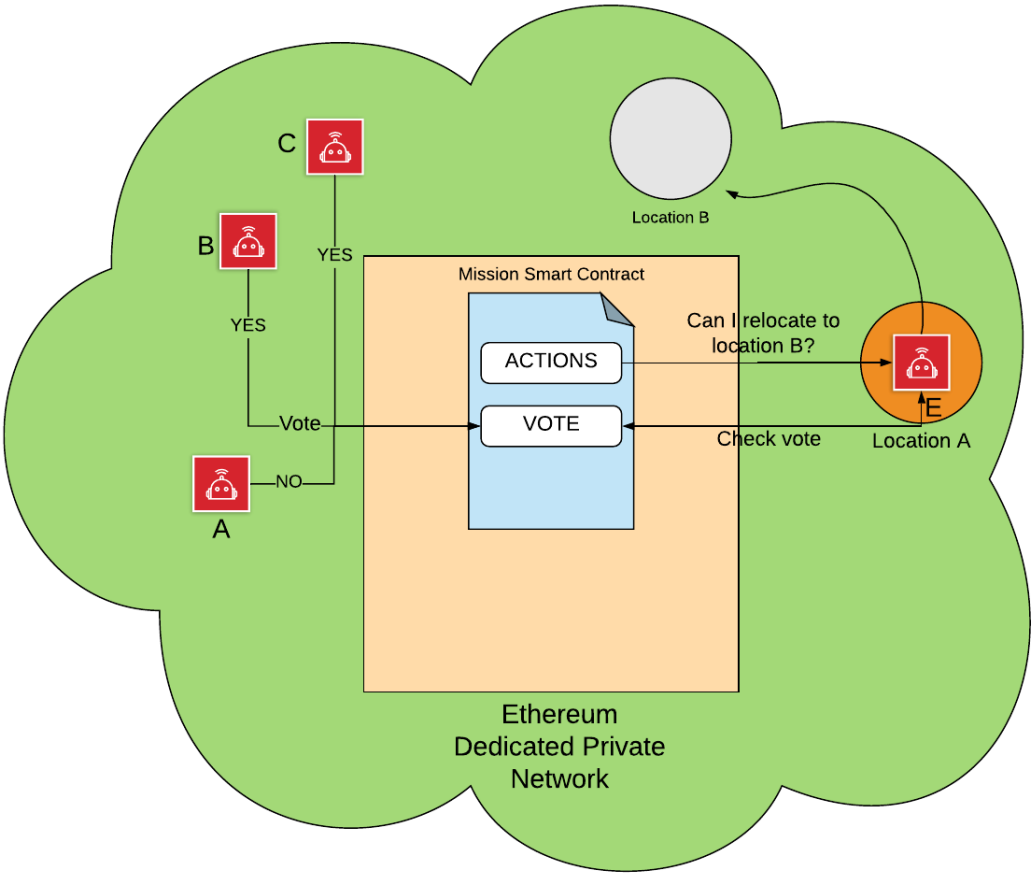

Fig. 8. Multi signature overview.

4.7 System Flow

In this section, the main system events are described to give an in-depth understanding of the system's life cycle. In the first step, all robots and users must be registered in the system. Only authorized robots (registered) are capable of providing services. On the other hand, only authorized users (registered) are capable of requesting services. Each user surveillance mission request initiates a surveillance mission smart contract deployment on a dedicated private blockchain. Then, robots join the

surveillance mission based on the user request and the control center order. The robots join a dedicated mission private blockchain to start the surveillance mission that is requested by the user.

4.7.1 User and Robots Registration

Users and robots need to register in the system using the control center smart contract. All of the registration data is stored on the smart contract. As mentioned before, only registered users are authorized to request a surveillance mission. Only registered robots are authorized to join and participate in a mission. A surveillance mission smart contract is deployed on the Ethereum private blockchain. Robots automatically register using the running daemon code. Fig. 9 shows the overview of registration.

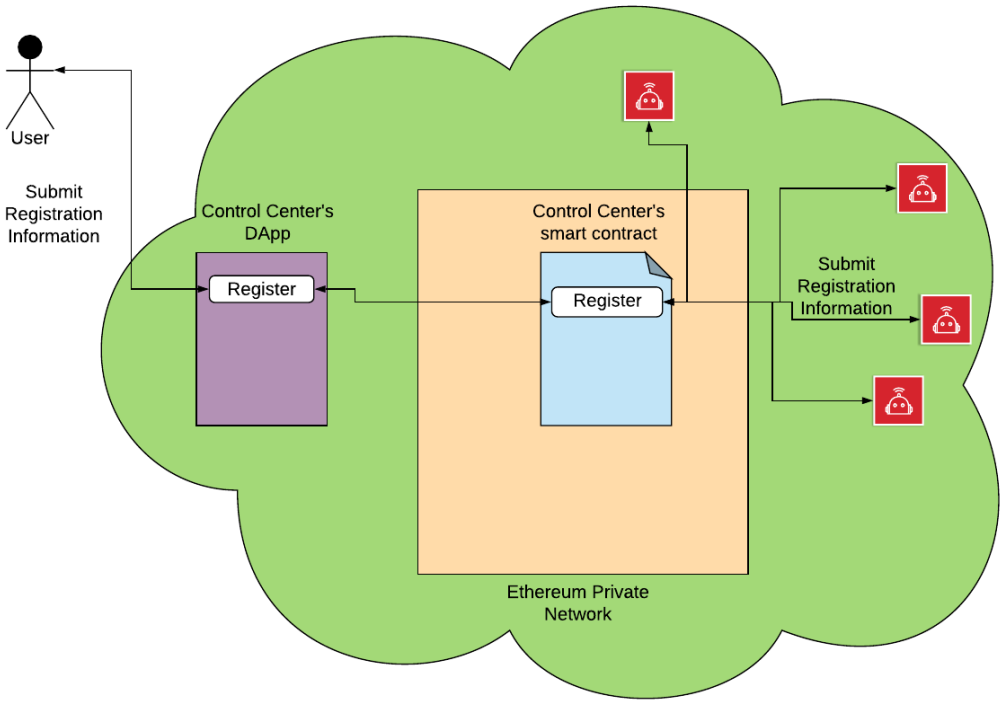

Fig. 9. Swarm registration.

Users and robots need to provide different information at the time of registration. Robots need to provide blockchain address and geolocation. Users need to provide blockchain address, username, the mission location (providing boundaries).

4.7.2 Surveillance Mission Smart Contract's Deployment

The control center DApp provides the capability for the users to deploy a surveillance mission on the Ethereum private blockchain. The users are the only actors in the system who can initiate surveillance mission service requests. As mentioned in 4.7.1, only registered users are authorized to request a service. Fig. 10 shows the overview of the steps taken to deploy a surveillance mission smart contract.

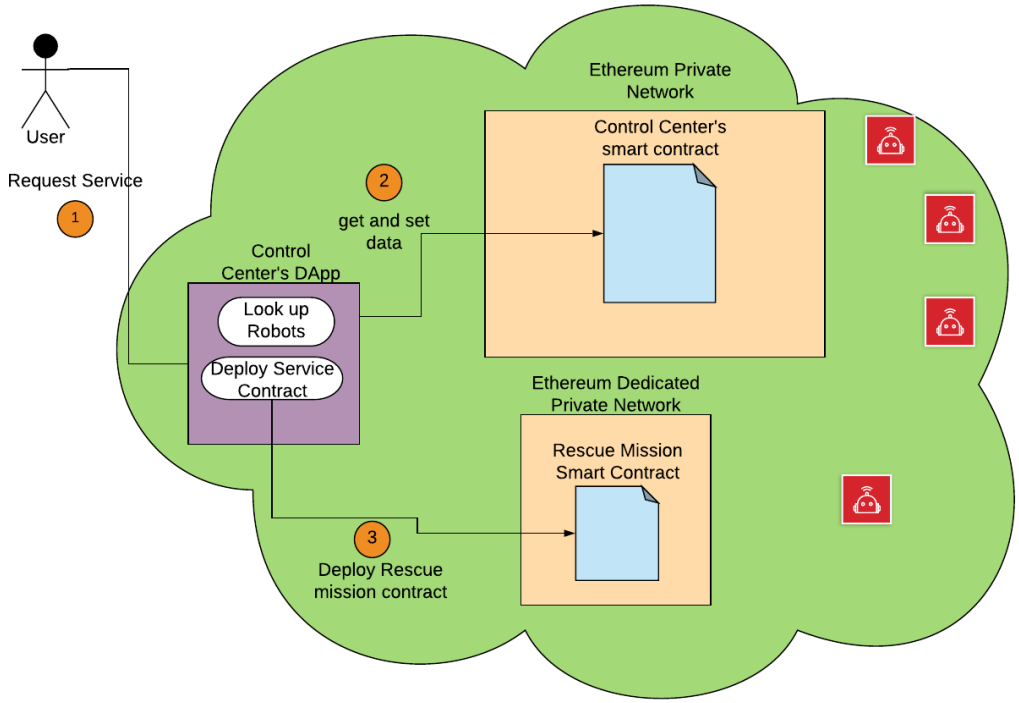

Fig. 10. surveillance mission deployment.

Each surveillance mission contract is deployed with the required information. In stage one of the process, a user requests a surveillance mission service based on the information provided at the registration point. In stage two, the control center DApp interacts with the control center smart contract to check if the required number of robots are available for the specific mission. In stage three, a surveillance mission contract is deployed on a dedicated Ethereum private network if the mission requirements are met. The requirements of a mission are the robots' location and charging rate. The mission contract is deployed on a dedicated private blockchain. The mission deployment on a dedicated private network eliminates the unnecessary PoW computation by the swarm. Also, the blockchain ledger is dedicated only to the swarm's activities. The surveillance mission contract is constructed by passing a list of the service provider robots.

4.7.3 Robots' Blockchain Node Deployment

Each robot executes a Geth client when it is activated. The control center activates a robot if the robot is available and fits a surveillance mission criteria. Below is the control center smart contract activation method:

```
function activateRobot(address robot_address) public returns (bool){
        if (msg.sender == admin){
            robots_activated[robot_address] = true;
            return true;
        }
        return false;
    }
```

A robot is constantly checking its activation status. Once the robot's activation flag is set, the robot initiates a Geth client to create a swarm by connecting to all other activated robots in a dedicated mission private blockchain. Fig. 11 shows the flow chart of this step.

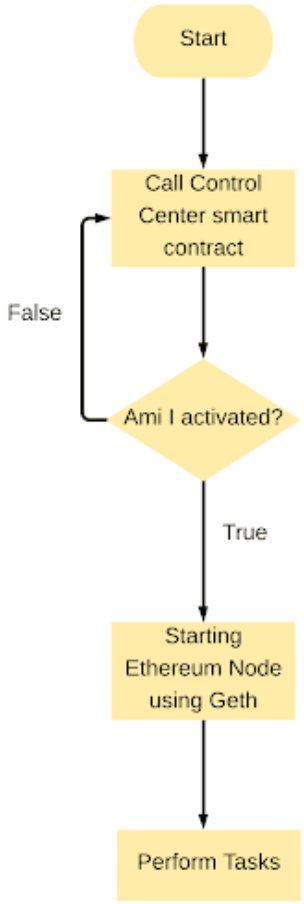

Fig. 11. A Robot activation flow.

4.7.4 Surveillance Mission

Robots are constantly updating their information after being activated. Robots' Information is updated using transactions sent to the mission's smart contract. Each robot sends transactions to the private blockchain network. Fig. 12 shows the overview of information updates by robots by storing them in a surveillance mission smart contract.

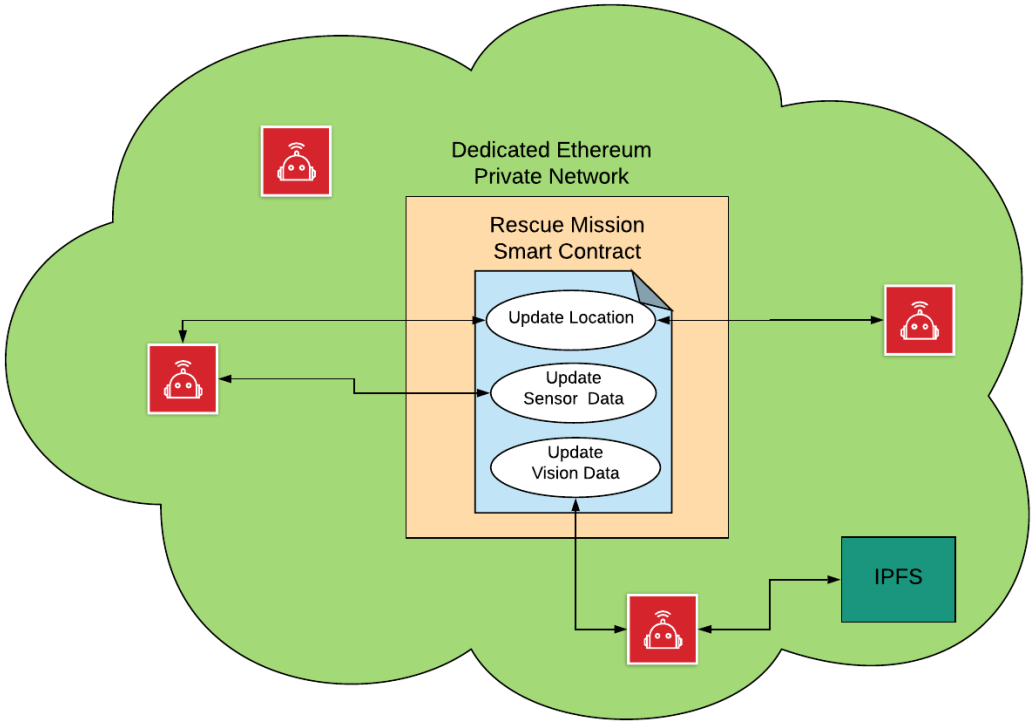

Fig. 12. Robot's activation flow.

Each robot running a Geth client is capable of communicating with the blockchain. Each robot updates its location and speed by sending transactions to the blockchain network. Each robot in swarm collects data by using its sensors and vision capability. In the event of finding the desired surveillance information, a robot stores the data in the mission smart contract. Vision data (an image) is updated in two steps. First, the data is uploaded to the IPFS network. Then, the returned hash value is stored on the smart contract. A user can retrieve the data by using the control center DApp.

5 EVALUATION

The focus of this section is to evaluate the usability and application of the existing Ethereum blockchain with the PoW consensus in a swarm of robots. Performance, scalability, recoverability, and responsiveness of the system are evaluated.

5.1 Test Setup

All tests and evaluations are performed on an Intel(R) Core(TM) i7-7700 CPU 3.60GHz (x86 64 bit) Linux (Debian stretch) 9.6 with four cores (8 CPU) and 16GB RAM. Virtual Robot Experimentation Platform (VREP) is used for the swarm's simulation. VREP is based on a distributed control architecture. Fig. 13 shows the virtual environment created by VREP.

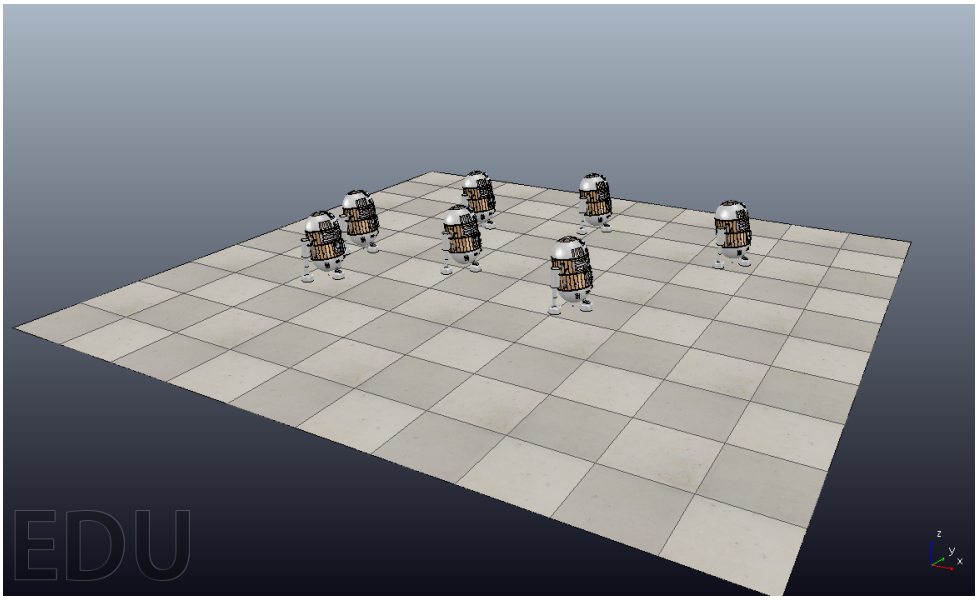

Fig. 13. Virtual robot experimentation platform.

As shown in Fig. 14, Kubernetes is used for container orchestration. Each robot is a docker container (process) in the testing environment. Each robot system's image is custom built to provide the capabilities discussed in 4.2. Each robot sends commands to

the VREP simulation environment's controllers for the simulation of its movements and actions. In addition, a Redis server is used to orchestrate the swarm's testing.

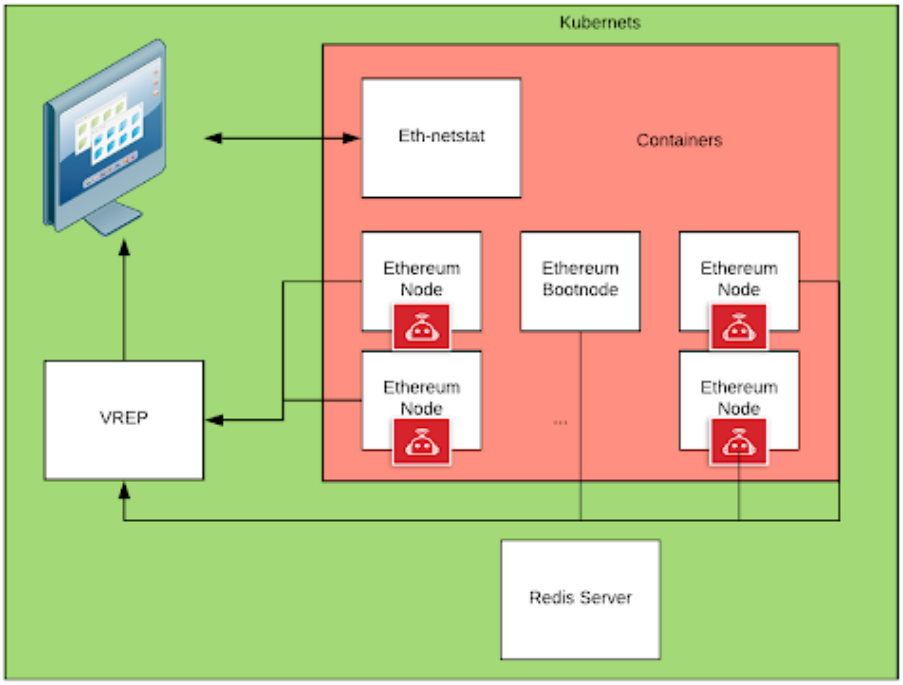

Fig. 14. Testing setup overview.

5.2 System Evaluation

The system evaluation was performed to answer one question: Is it feasible to use the Ethereum blockchain in a swarm robotics system to perform the surveillance mission? In each test, the CPU usage of each robot is monitored. The goal is to find a threshold at which a robot can perform all of the tasks mentioned in 4.2. Each test is conducted in a swarm of different sizes. The Linux machine mentioned in 5.1 can run at most nine Geth clients at the same time. As a result, the minimum size of swarm being tested is three and the maximum size is nine.

5.2.1 Performance

As mentioned in section 2.1.5.2, each interaction with blockchain is either a call or a transaction. Transactions need to be confirmed and verified by all of the blockchain network participants (miners). First, the robots' CPU utilization was monitored to find the ideal specifications for robots. An ideal specification is the minimum setting that a robot needs to be able to perform all of the tasks mentioned in 4.2. This test was performed in a swarm of size five. Each robot commits ten transactions in each test case. Also, each robot has ten percent of the total memory (16GB total). The time required to commit and verify all of the transactions is monitored. The CPU percentage represents the utilization of each computer core (eight cores, 3.60 GHz). Fig. 3 shows the result of these tests.

Table 2
System Performance Test

CPU %	**Memory %**	**Avg Block Time**	**Avg Transaction per Block**	**Time**
5	10	9.2	0.17	362.23
10	10	7.5	0.36	149.43
15	10	3.4	0.55	104.98
20	10	2.39	0.89	83.71
40	10	2.46	0.92	38.66
60	10	2.42	2.4	23
80	10	2.46	2.3	29.19
100	10	2.43	3.57	15.36

Next, this study measured the required time for robots' Geth deployment and peer-to-peer connectivity. After running a Geth client, a robot becomes a node in a swarm's blockchain network. A normal state is defined as a state in which robots are fully synced and connected. There are three steps required before starting the time measurement. First, a user and robots need to be registered in the system (4.7.1). Second, users request a mission (4.7.2). Third, robots activate a Geth client (4.7.3). The timing starts from the point that all of the robots' activation flags are set to true, and it

ends when all of the robots are connected. Fig. 15 shows the result of the timing of the Geth client execution by swarms of different sizes.

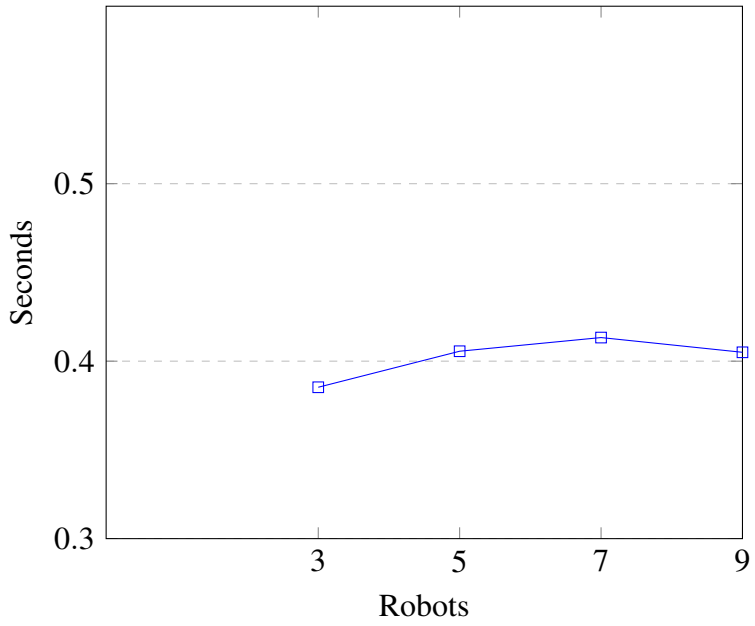

Fig. 15. Time taken for all Geth clients to run.

Fig. 16 shows the result of the required time for each swarm to be fully connected. It can be said that a swarm is fully connected when all of the peers are connected.

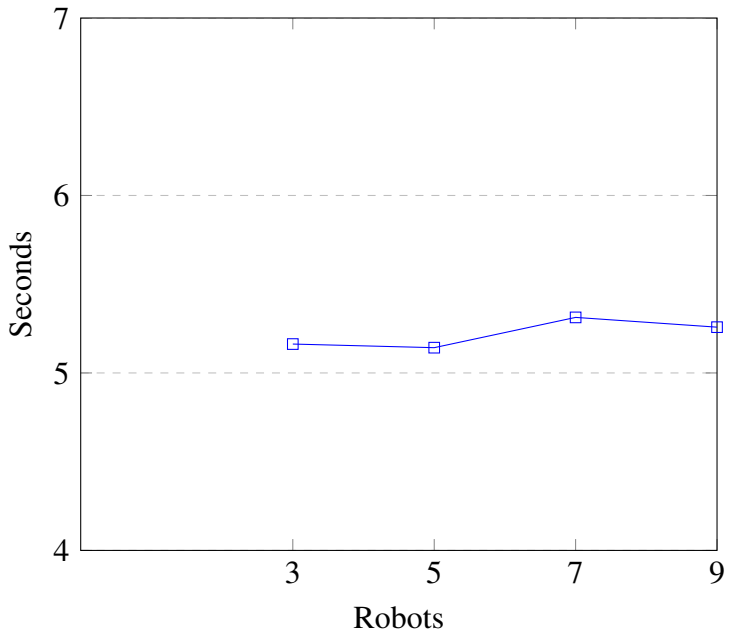

Fig. 16. Time taken for all peers to connect

As mentioned in section 4.7.4, the robots are constantly updating their information by committing the transactions to the blockchain's surveillance mission smart contract. Table 3 shows the performance test results of the robots' information updates. This stage starts once all the robots have a Geth client running locally, and it ends when the surveillance mission is finished. In this stage, each robot is actively moving around in an assigned area. An area is assigned to each robot at the time of a mission deployment. The "setRobotMetadata" is the smart contract's method used by robots to update their information. Robot's Speed, longitude, and latitude are passed as the transaction arguments in every robot's information update.

The Average Duration column is the total time that a swarm needs to commit and confirm all of the transactions. The average time of a block verification by all miners was about 1.1 second. Each block contained an average of 4.24 transactions.

Table 3
Transactions Performance Test at Initiation Stage

Swarm Size	Each Robot Transaction Count	Average Duration in minutes
3	300	1.16
3	3000	10.86
3	30000	115.6
5	500	1.65
5	5000	16.03
5	50000	187.2
7	700	2.58
7	7000	21.5
7	70000	250.8
9	900	4.1
9	9000	26.4
9	90000	307.7

5.2.2 Scalability

As mentioned in section 2.1.5.2, each full node stores a local copy of the blockchain ledger. Since robots do not have an unlimited storage space, storage becomes a critical metric to measure. The storage examination starts when the first transaction is committed, and it ends when all of the transactions are confirmed. The "setRobotMetadata" method is used as well for testing storage scalability. The test is performed on a swarm with five robots. The result is shown in Table 4.

Table 4
Transactions Scalability

Total Transaction Count	Average megabytes added to the ledger
500	1.1
5000	13.8
50000	122
500000	1302
5000000	12422
50000000	129000

As stated in section 4.1.3, two types of Ethereum nodes are used in a swarm. Change of node types within a swarm doesn't change the storage utilization. Using light nodes within a swarm only puts more computation and storage load on the swarm's full nodes. If we assume that the block time is two seconds and each block contains an

average of four transactions, then we can say that a minimum of four Gigabyte is added to the ledger in twenty-four hours.

5.2.3 Recoverability

Applications crash for various reasons which can be out of the control of the developers [19]. Crashes and failures such as power outage and network congestion can happen at any point of a swarm's life cycle. Recoverability testing is evaluating the recovery time of a swarm after a failure. In this test, a swarm starts in a normal state. A normal state can be defined as the state in which all of the robots are connected using peer-to-peer over TCP connection (RPLx). A crash is simulated in a swarm by shutting down and disconnecting a robot from the network. Variable T can be defined as the time interval (seconds) that a robot is disconnected from the network. After T time, a disconnected robot rejoins the swarm by running the Ethereum Geth client. Variable RT is recovery time in seconds. Recovery time is a time window (seconds) that a robot needs to recover from a crash. It can be said that a robot recovered from a crash once the following criteria are met again:

- A Geth client is running
- The robot connected to the blockchain network
- The robot is connected to all peers
- The robot is synced with all peers

The result of recoverability is shown in Table 5.

Table 5
Recoverability Test

Swarm Size	**Missed Blocks**	**Recovery Time**
3	1000	14.473
3	10000	45.64
3	100000	310.25
5	1000	18.4
5	10000	51.2
5	100000	322.1

5.2.4 Responsiveness

In this test, responsiveness refers to the ability of a swarm to response to a signal sent from one of the robots or the contract's owner (user). A robot or an outside authority can send a signal to the swarm at any time. The signals are used to provide global knowledge to a swarm. For example, if there is an emergency, a robot can send a signal to the swarm so the swarm can take the required actions. In this test, we measure the time taken for all robots to acknowledge a signal. The test starts from the moment that a signal is sent to the swarm and it ends when all of the robots know about it. A signal is a simple flag that is set in the surveillance mission smart contract by calling the "setFlag" method. Table 6 shows the result of responsiveness measurement in seconds.

Table 6
Responsiveness Test

Swarm Size	**Responsive Time (RST)**
3	2.169
5	3.705
7	1.79
9	5.75

6 DISCUSSION AND FUTURE WORK

As shown in section 4, the Ethereum blockchain can provide a platform for swarm robotics systems. Robots can effectively communicate and perform their assigned tasks with the help of blockchain. As shown in section 4.7, robotic tasks can be replicated into a blockchain-enabled swarm. Although the Ethereum blockchain provides a usable infrastructure for swarm robotics, it has performance inefficiencies shown in the results in section 5.2.

As shown in 5.2.1, the minimum of 0.36GHz (80% of one core) of the processing power is needed to prevent a long data propagation delay in the network. Also, as shown in 5.2.1, the memory (RAM) utilization of at least 1024 megabytes is required for a robot to operate normally. With such a utilization, a swarm can confirm and propagate the data without a noticeable delay. The results of the performance test show that swarms are heavily impacted by the amount of CPU allocated to each robot.

However as shown in 5.2.1, a swarm's startup and connectivity take an average of 5.2 seconds, if the robots have at least 0.36GHz (80% of one core) of processing power. In addition, as shown in 5.2.4, the system acknowledges a signal sent to the network at an average of 3.35 seconds. The required time depends on the blockchain network load at the time of the signaling.

Considering all of the results, a Raspberry Pi 3 Model B+ could be an ideal hardware candidate for a robot in a blockchain-enabled swarm. The Raspberry Pi 3 Model B+ fulfills the wireless capability required by a robot. In addition, it has 1 GB of memory (RAM) and 1.4GHz CPU which allows it to perform all of the necessary PoW consensus calculations.

The PoW consensus algorithm is not feasible for robots that need to participate in surveillance missions that require more than seven days. As a mission continues, the required storage increases for each robot in the swarm (shown in 5.2.2).

Developers and companies are continually working on new algorithms to make the blockchains' participation feasible for all devices—regardless of computation power and storage size—to be able to join and participate in a blockchain network. As mentioned in section 2.1.3, the Ethereum is already working on the implementation of the PoS. Also, the practicality of using other consensus algorithms like delegated proof of stake, proof of authority, proof of weight should be explored. The evaluation of the proposed swarm robotics system should be executed with real robots and complex traditional swarm robotics algorithms. This in-depth evaluation will provide better direction on how to fix the performance shortcomings in swarm robotics systems.

7 CONCLUSION

Swarm robotics is an emerging technology in which a swarm of simple robots collaborate and communicate to accomplish a complex task. Swarm robotics is facing limitations in autonomy and security due to the missing global environmental awareness within a swarm. Blockchain technology has been able to create decentralized and distributed systems. It demonstrates the capability of reaching a consensus within a group of distributed agents using the secure asymmetric cryptographic algorithms. This paper presented the usability of blockchain in swarm robotics systems by demonstrating a surveillance mission use case. In this work, a simulated swarm was tested to determine if the Ethereum blockchain with PoW can be an ideal framework for swarm robotics systems. The results show the blockchain eliminates a single point of failure, provides global knowledge, and makes a surveillance mission possible. However, storage limitation and inefficiency of the PoW consensus algorithm do not allow robots to operate in a long surveillance mission due to high CPU and memory utilization. As better consensus algorithms are being developed, blockchains can become more efficient at the required processing and computation power. As a result, blockchain technologies like Ethereum can become an ideal platform for swarm robotics systems since they provide fault tolerance, security, and global knowledge by design.

Literature Cited

[1] U. Grohmann, "Paint robots in the automotive industry–process and cost optimization," *Industrial Robot: An International Journal*, vol. 23, no. 5, pp. 11–16, 1996.

[2] D. Evans, "The internet of things: How the next evolution of the internet is changing everything," *CISCO white paper*, vol. 1, no. 2011, pp. 1–11, 2011.

[3] M. Gupta, D. Saxena, S. Kumari, and D. Kaur, "Issues and applications of swarm robotics," *International Journal of Research in Engineering, Technology and Science*, vol. 6, pp. 1–5, 2016.

[4] E. C. Ferrer, "The blockchain: a new framework for robotic swarm systems," in *Proceedings of the Future Technologies Conference*, pp. 1037–1058, Springer, 2018.

[5] T. Laurence, *Blockchain for dummies*. John Wiley & Sons, 2017.

[6] M. Hölbl, M. Kompara, A. Kamišalić, and L. Nemec Zlatolas, "A systematic review of the use of blockchain in healthcare," *Symmetry*, vol. 10, no. 10, p. 470, 2018.

[7] J. W. Bos, J. A. Halderman, N. Heninger, J. Moore, M. Naehrig, and E. Wustrow, "Elliptic curve cryptography in practice," in *International Conference on Financial Cryptography and Data Security*, pp. 157–175, Springer, 2014.

[8] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[9] V. Buterin *et al.*, "Ethereum white paper," *GitHub repository*, pp. 22–23, 2013.

[10] I. Navarro and F. Matía, "An introduction to swarm robotics," *Isrn robotics*, vol. 2013, 2012.

[11] N. Correll and D. Rus, "Architectures and control of networked robotic systems," *Handbook of collective robotics: fundamentals and challenges*, pp. 81–103, 2013.

[12] A. Banafa, "Three major challenges facing iot," *IEEE Internet of things*, 2017.

[13] K. Korpela, J. Hallikas, and T. Dahlberg, "Digital supply chain transformation toward blockchain integration," in *proceedings of the 50th Hawaii international conference on system sciences*, 2017.

[14] D. Galvin, “Ibm and walmart: Blockchain for food safety,” *IBM & Walmart*, 2017.

[15] J. Ackermann and M. Meier, “Blockchain 3.0 - the next generation of blockchain systems,” 09 2018.

[16] F. Mondada, G. C. Pettinaro, A. Guignard, I. W. Kwee, D. Floreano, J.-L. Deneubourg, S. Nolfi, L. M. Gambardella, and M. Dorigo, “Swarm-bot: A new distributed robotic concept,” *Autonomous robots*, vol. 17, no. 2-3, pp. 193–221, 2004.

[17] J. Benet, “Ipfs-content addressed, versioned, p2p file system,” *arXiv preprint arXiv:1407.3561*, 2014.

[18] D. Yaga, P. Mell, N. Roby, and K. Scarfone, “Blockchain technology overview,” tech. rep., National Institute of Standards and Technology, 2018.

[19] E. J. Braude and M. E. Bernstein, *Software engineering: modern approaches*. Waveland Press, 2016.