TI-Share: Sharing Threat Intelligence using Structured P2P and Blockchain

CS 4675-6675 Project Proposal
February 21, 2025

Keerthana Thotakura	kthotakura3@gatech.edu	903790473
Maria Jothish	mjothish6@gatech.edu	903754696
Sabina Sokol	ssokol3@gatech.edu	903863248
Tran Ha	tha46@gatech.edu	903719409
Seung-a Baek	seungabaek@gatech.edu	7709278360

Table of Contents

Motivation and Objectives

Whether you are clicking on a search result, creating a new online account, or booking a hotel, you rarely consider the serious consequences a security breach in that system or website application can cause. With a quick tap and a few answered questions, your information is stored somewhere unknown to you, yet you are more focused on the fact that you've gained access to new information or service. Along with the advancement of technology, it seems like our sense of trust in the security measures put in place on these applications has also increased. However, whether it be a government agency, education institution, or a public/private business you are dealing with, all these organizations are vulnerable to security breaches.

Well-known organizations like Equifax, Yahoo, eBay, Facebook, and even Marriott Hotels have faced severe breaches resulting in millions, if not billions of personal data being compromised. Imagine going on a relaxing 2-week vacation only to return and realize your identity has been stolen. That was exactly the case with Marriott Hotels, except they discovered their security breach 2 years after it occurred. Therefore, customers only realized 2 years later that their credit card information, passport numbers, and personally identifiable information was disclosed to an unknown attacker [1]. Thus, now only may a customer realize why their credit card got compromised 1 year back. Now to those 500 million customers, gone are the days of security and privacy of their information.

With the various kinds of malware, hacking, and cyber-attacks that arise daily, it is unfair to assume that this security breach was preventable. While an organization may build a firewall that is seemingly secure with all the latest security features and endless testing to find areas of weakness, a new curveball attack could dismantle it in a matter of seconds. In fact, it might be fair to say that the presence of these threats is inevitable since we don't live in a utopia and as a system gets more complex, it gets less secure [2]. However, what can be done on our part is to find a method that can potentially reduce the amount of time it takes for network security professionals to identify a breach in a secure manner, resulting in as minimal damage as possible for users/customers of the application/organization. Thus, our group proposes TI-Share to efficiently and securely identify threats.

The goal and objectives of this project is to build a peer to peer network based threat sensing system. Current solutions are cloud based or centralized threat sensing systems which are weak as they are prone to malware. When dealing with cybersecurity threats and malware efficiency and speed in knowing they exist and crucial in dealing with them in a timely manner. Our proposed solution is building a decentralized P2P system that way we can make a threat sensing system that is more resilient and fast. The overall objective is to build a more efficient and secure decentralized P2P threat sensing system. The goal is to build a system that is more efficient, fast, and secure than current solutions and we plan on achieving this by utilizing blockchain technology to authenticate the information being shared and leveraging a P2P network rather

than cloud solutions which would help increase the speed and scalability issues of current solutions.

The scope of this project is to build a decentralized network, a P2P network that will share cybersecurity issues and threat information across peers in real time while simultaneously authenticating the information being shared using block chain technology. We will be leveraging existing open source blockchain and cybersecurity threat analysis technologies to build the P2P network as well. People who have concerns regarding malware threats can leverage this network to more effectively share information about new threats and deal with them promptly. The user community would be IT teams and threat detection teams who would be able to leverage this tool.

Related Work

Past work on P2P networks includes centralized solutions and decentralized architectures like Gnutella. With a centralized solution, a centralized authority conducts further exploration and stores information about the data that is being collected in one spot. The issue with this is it can lead to link congestion, being a single point of failure, and expensive administration [3]. Congestion is highly likely when our agent/peer on the network is constantly logging network activity and then having to send it to its neighbors. Also, as our network grows, the efficiency at which logs are sent out will decrease, which isn't favorable for our purposes as we are trying to reduce the time it takes to identify a threat/breach. In addition, the vulnerability of easily being a single point of failure would defeat the purpose of our project because if the *one* agent we have logging network activity fails, then there would be no way for network professionals to view network traffic.

Another aspect we need to take into consideration is what if an attacker accesses and modifies the packet meta data our agent/peer is collecting. This means the attacker has breached the data in our network, but our peer couldn't detect this anomaly because the attacker has also modified the network log our peer is collecting in a way that makes it seem like nothing unusual is going on in the network. In such a scenario, since we have only one peer collecting this metadata, if an attacker changes it, we will never know since there is no other peer also collecting metadata on the same network that we compare against for discrepancies. Ultimately, this allows the attacker to continuously extract data from the network. Therefore, due to the aforementioned issues when utilizing a centralized P2P architecture, we decided to incorporate a decentralized P2P system instead.

By using a decentralized system, multiple peers on our network will be splitting up what network activity they collect and then validating what they've collected with other peers that collected the same data. This way we don't have to worry about link congestion since all of the peers won't be interacting with each other. Also, since there are multiple peers collecting the same network meta data, we don't have to worry about having a single point of failure because there are multiple copies. Having multiple copies also resolves the issue of being unable to detect anomalies due to the log of one peer being manipulated since we can just compare it with another peer that collected the same data and identify the discrepancy. An example of a decentralized P2P system most relevant to our project is Iris.

Iris is the first global P2P system created for threat intelligence sharing. Most P2P systems are created for file sharing, storage, chat, etc. but they are not prepared to share security detection, and threat intelligence data and alerts [4]. Iris is a completely decentralized P2P network that allows peers to (i) share threat intelligence files, (ii) alert peers about detected attacks, and (iii) ask peers about their opinion on potential attacks [5]. When a peer is attacked it automatically shares threat intelligence data and alerts the network of a new attacker. It also allows peers to ask about the reputation of other peers and defines

organization in the network by using DHT and private/public keys thus handling the problem of confidentiality. By having each organization cryptographically-verified, peers can only send content to pre-trusted groups of peers. Peers on the network have control of the privacy of their data by fixing what organizations or peers can have access to it and they can also control the transfer of data by using epidemic algorithms. Although our projects are similar in that both intend to minimize the damage threats/breaches do on a network, what separates our project from Iris is that our project focuses more on quickening and securing the process it takes to identify a threat rather than quickening the pace information regarding a threat is shared among other peers in the network.

Proposed Work

Our project will be focusing on developing a peer-to-peer-based threat intelligence network. Unlike any other traditional centralized threat intelligence systems that rely on cloud-based updates, our system will distribute anomaly activity logs across a decentralized network of nodes. This will help quickly identify new cyber threats. The main system will consist of a peer-to-peer network, blockchain based data verification with storage, and an interactive dashboard. It will be structured so that the system will be secure and efficient.

Design and implement a P2P network:

The system will be designed using a distributed file sharing protocol to support peer-to-peer communication between nodes. Each peer will detect and report threats and any anomalies, preventing any malicious file spreading. To optimize it, a gossip protocol or distributed message passing model will be implemented that helps with network congestion. Multiple nodes will verify logs by cross-referencing with other peers, ensuring that the information is correct and not fabricated.

Develop blockchain storage and verification:

The blockchain components will ensure the integrity, transparency, and authenticity of shared intelligence. There are several types we can use, from permissioned blockchain to lightweight distributed ledgers technology, to securely log threat reports. Every new detection will be cryptographically hashed and stored on the blockchain so that it is tampered proof.

We can use a hybrid model to minimize computational overhead and storage constraints:

- On chain data storage: It will store only the hash references of logs directly on the blockchain. This will ensure that the data is secure and immutability.
- Off chain data storage: This will keep full log details and raw data for efficient retrieval.

We can also implement a lightweight consensus mechanism to help with processing verification faster without excessive computation costs. Smart contracts can be used in the enforcement of access rights, confirming shared intelligence, and protecting against unauthorized alterations.

Frontend Dashboard

We will include an interactive dashboard using a modern front-end framework to allow users to view threat intelligence data in real time. This dashboard may consist of:

- Logs of network activities
- Validation status from blockchain records
- Number of active peers
- Analysis and historical trends

Our primary users include:

- Government organizations
- Network security professionals
- Security analysts
- Incident Response Teams

Plan of Action - Design

Instead of relying on a single monitoring entity to then disseminate threat intelligence to other entities, we propose a tool that will leverage a blockchain or distributed ledger technology to create an immutable and decentralized log of network activity, which would ensure integrity and transparency of the decentralized network monitoring tool. The network logs would be cryptographically signed and validated by multiple participants in the system which further ensures data integrity. The architecture consists of distributed nodes that collaboratively monitor network traffic, detect anomalies, and store event logs in a verifiable and auditable manner, possibly in a lightweight database.

The functional components include:

1. Network Traffic Collection Agents

   These entities will be deployed on multiple P2P nodes within a network and capture real-time packet metadata (source/destination IP, timestamps, protocol type, volume, etc). They will also perform basic anomaly detection (detecting unusual spikes in traffic, unauthorized access attempts, etc) and send logs to the validation layer of the blockchain/DHT system for verification and storage.

2. Blockchain Log Storage

   We will use a permissioned blockchain (Hyperledger Fabric, Corda, etc.) to ensure that logs are tamper-proof and verifiable, and these logs will then be stored as hashed transactions to reduce storage size while maintaining integrity. We are also considering implementing smart contracts or chaincode enforce access policies (who can view/edit logs, for example).

3. Peer Validation & Consensus Mechanism

   The nodes participating in the system will validate incoming logs before they are appended to the ledger, and we will use a lightweight Proof of Authority or Byzantine Fault Tolerance consensus mechanism to ensure rapid validation without high computational overhead, which is an essential limitation to our design so that our tool is attractive to potential users as a lightweight option for enhanced network security. Each log entry must then be approved by a minimum number of validating peers.

4. Anomaly Detection & Alerting System

   We will develop a decentralized AI-powered analytics engine or other open-source network analysis system to detect suspicious patterns across the network. Alerts will be generated when anomalies are detected, and other peers in the network will be notified. The system can compare current traffic patterns against historical data stored on the

ledger, a feature of blockchain or DLT that makes it critical to the validation and authenticity component of our tool design.

5. User Interface and API Gateway

   We will develop a web-based dashboard for users to visualize network activity, flagged anomalies, and ledger transaction history. We will also potentially include RESTful API gateway to allow third-party applications to query the ledger for forensic analysis and compliance audits. Admins and security teams would also be able to drill down into specific logs to investigate incidents.

Other technical considerations include:

1. Scalability & Storage Optimization

   Full logs won't be stored on the blockchain directly (only hash references), while raw data is kept in an off-chain distributed storage (IPFS, BigchainDB, etc.). Nodes will use a lightweight, efficient blockchain implementation to reduce overhead and state bloating (continuous growth as data, smart contract information, etc. are stored).

2. Privacy & Access Control

   Logs will be encrypted using Zero-Knowledge Proofs or Attribute-Based Encryption to ensure privacy while maintaining verifiability. Role-based access control will be used to determine who can access certain logs.

3. Integration with Existing Tools

   We can interface with existing Security Information and Event Management solutions like Splunk, ELK Stack, etc, and standard network monitoring protocols (NetFlow, sFlow, etc.) for compatibility.

4. Deployment Models

   We can either have an on-premises model for organizations that need full control over data, a cloud-hosted with decentralized nodes running in different regions, or a hybrid deployment with edge nodes collecting traffic and sending summarized logs to the blockchain.

Overall, the key benefits of this tool with this P2P and blockchain based implementation include:

1. Tamper-proof logging, which eliminates risks associated with manipulated network logs.
2. Decentralized security such that there is no single point of failure or centralized authority.
3. Transparency where every action is verifiable and traceable.

4. Anomaly detection and alerting that enables proactive security insights across distributed networks.
5. Efficient compliance tracking to help in meeting regulatory security standards (GDPR, NIST, SOC 2).

Essentially, our basic project steps are deciding on a blockchain framework, prototyping a network monitoring agent, integrating blockchain storage, developing a web UI for log analysis, and testing network scalability and validation speed.

Division of Work

Blockchain Validation: Seung-a

Blockchain Storage: Tran, Maria

P2P Networking Agent: Sabina, Keerthana

Web UI: Everyone in a later phase once we have a better understanding of how the system works

Project Timeline

Week 1 (Feb 24 to Mar 2)

- ☐ Proposal meeting with professor
- ☐ Finalize problem statement, scope, and technical approach
- ☐ Research and select the blockchain/DLT framework to be use so that Blockchain Validation and Storage subgroups can begin work in sync
- ☐ Research and select basic networking monitoring approach that integrates with P2P framework
- ☐ Set up Github repository and project management board (Trello)
- ☐ Agree on cadence of team meetings and check-ins with TA

Week 2 (Mar 3 to Mar 9)

- ☐ Implement basic network monitoring and P2P network
- ☐ Define initial data schema for logs
- ☐ Start setting up blockchain/DLT environment

Week 3 (Mar 10 to Mar 16)

- ☐ Set up smart contract/ledger transaction structure for blockchain/DLT validation
- ☐ Implement basic log storage mechanism on blockchain
- ☐ Establish validation/consensus mechanism for blockchain
- ☐ Test writing sample logs to blockchain

Week 4 (Mar 17 to Mar 23)

- ☐ Develop peer validation process for entries (P2P side)
- ☐ Implement cryptographic hashing for logs on blockchain for data integrity
- ☐ Implement off-chain storage for efficient log storage
- ☐ Begin presentation slides for workshop

Week 5 (Mar 24 - Mar 30)

- ☐ Implement basic UI for viewing logs and alerts
- ☐ Set up REST API for querying blockchain logs
- ☐ Integrate basic analytics/anomaly detection tool
- ☐ Conduct initial testing

Week 6 (Mar 31 to Apr 6)

- ☐ Refine alerting/anomaly feature
- ☐ Finish up presentation slides for workshop

- [ ] Conduct further system testing

Week 7 (Apr 7 - Apr 13)

- [ ] Workshop presentation
- [ ] Get feedback from professor
- [ ] Make adjustments based on feedback
- [ ] Conduct performance testing/develop full usability evaluation sequence

Week 8 (Apr 14 to Apr 20)

- [ ] Prepare demo script/test cases
- [ ] Conduct mock demo runs
- [ ] Finalize documentation/user guide

Week 9 (Apr 21)

- [ ] Final project demo!

Evaluation and Testing Method

For the MVP, our primary goal is to verify whether threats are successfully propagated across the network in a simple yes/no (boolean) manner and are securely recorded on the blockchain.

For testing, we will introduce real-world malware, phishing URLs, or malicious IPs in a sandboxed environment to check how effectively the system detects and propagates these threats.

Our evaluation criteria at this stage will be:

a. A node is able to submit a log to the network indicating the presence of a threat if applicable.
a. Other peers are able to receive the log and verify its presence.
b. A peer can query the network to check if a previously submitted log exists.
c. The system successfully records a log entry, ensuring it remains retrievable.
d. A node can verify that a log has not been tampered with by checking the blockchain record.
e. A user is able to view logs from the network in real time on a front-end platform.

The ideal stretch goals of our project if it was continued outside of the scope of our course would have these metrics:

Evaluation Area	Metrics	Evaluation Model
Threat Detection Accuracy	% of correctly detected threats	Simulate known threats and measure detection rate, how many false positives? False negatives?
Threat Propagation Speed	Time taken for a detected threat to be added to a log and shared across the network	Insert a threat into one node and track how fast the log is updated and reaches other peers
Blockchain Validation Speed	Time taken for new logs to be verified and stored	Record time from submission to confirmation in blockchain
System Scalability	Maximum number of nodes that can participate without failure	Test with increasing number of nodes and measure performance degradation
User Satisfaction	% of users who find the system useful and intuitive (If we end up creating a front-end dashboard)	Conduct user testing and collect survey responses

Bibliography

[1] Tariq, Muhammad Usman. "Data breach incidents and prevention in the hospitality industry." *Corporate Cybersecurity in the Aviation, Tourism, and Hospitality Sector*. IGI Global, 2024. 181-199.

[2] Schneier, Bruce. "Software complexity and security." Mar. 2000.

[3] Apoidea: A Decentralized Peer-to-Peer Architecture for Crawling the World Wide Web, A. Singh, M. Srivatsa, L. Liu and T. Miller, In the proceedings of the *SIGIR workshop on distributed information retrieval*, August 2003.

[4] “NLnet; Threat Intelligence Sharing.” *Nlnet.nl*, 2025, nlnet.nl/project/Iris-P2P/.

[5] HappyStoic. “GitHub - HappyStoic/Iris: Iris - P2P System for Confidential Sharing of Threat Intelligence and Collaborative Defense for Slips.” *GitHub*, 2021, github.com/HappyStoic/iris?tab=readme-ov-file.