ASSIGNMENT 1

* Chord Protocol Implementation:

*Sample Input*:

m = 3

*Sample Output*:

Number of Bits to be allocated for search space (m): 3

Number of Nodes in the network: 0

Nodes in the network: No Nodes in the network!

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 1

Enter Node ID: 0

----------------------------------------------------------------

Number of Nodes in the network: 1

Nodes in the network: {0 , }

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 1

Enter Node ID: 1

----------------------------------------------------------------

Number of Nodes in the network: 2

Nodes in the network: {0 , 1 , }

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 1

Enter Node ID: 3

----------------------------------------------------------------

Number of Nodes in the network: 3

Nodes in the network: {0 , 1 , 3 , }

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 1

Enter Node ID: 5

----------------------------------------------------------------

Number of Nodes in the network: 4

Nodes in the network: {0 , 1 , 3 , 5 , }

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 1

Enter Node ID: 7

----------------------------------------------------------------

Number of Nodes in the network: 5

Nodes in the network: {0 , 1 , 3 , 5 , 7 , }

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 2

Enter Node ID: 7

----------------------------------------------------------------

Number of Nodes in the network: 4

Nodes in the network: {0 , 1 , 3 , 5 , }

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 4

i id + 2^i successor

Node 0 ->

0 1 3

1 2 3

2 4 5

Node 1 ->

0 2 3

1 3 5

2 5 0

Node 3 ->

0 4 5

1 5 0

2 7 0

Node 5 ->

0 6 0

1 7 0

2 1 3

----------------------------------------------------------------

Number of Nodes in the network: 4

Nodes in the network: {0 , 1 , 3 , 5 , }

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 3

Enter Node ID to search for: 4

Enter Node ID to begin search from: 1

Search Path: 1 -> 3 -> 5

----------------------------------------------------------------

Number of Nodes in the network: 4

Nodes in the network: {0 , 1 , 3 , 5 , }

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 3

Enter Node ID to search for: 6

Enter Node ID to begin search from: 0

Search Path: 0

----------------------------------------------------------------

Number of Nodes in the network: 4

Nodes in the network: {0 , 1 , 3 , 5 , }

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 3

Enter Node ID to search for: 6

Enter Node ID to begin search from: 5

Search Path: 5 -> 0

----------------------------------------------------------------

Number of Nodes in the network: 4

Nodes in the network: {0 , 1 , 3 , 5 , }

1. Join Network

2. Leave Network

3. Search File

4. Show Finger Tables

5. Close

Enter Choice: 5

* Security Issues of Chord Protocol:

The Chord Protocol is a distributed hash table (DHT) protocol designed for use in peer-to-peer (P2P) networks. It is primarily used for distributed data storage and retrieval in a decentralized manner. While Chord offers several advantages, such as scalability and fault tolerance, it also comes with its share of security challenges and issues. Here are some of the key security issues associated with the Chord Protocol:

1. **Data Privacy**: Chord does not inherently provide data privacy or encryption mechanisms. As a result, data stored within the Chord network can be vulnerable to eavesdropping by malicious nodes or unauthorized access. This issue can be mitigated by using additional encryption layers or secure communication protocols on top of Chord.
2. **Sybil Attacks**: Chord is susceptible to Sybil attacks, where a malicious node can create multiple fake identities within the network to gain disproportionate control over the network. This can lead to data manipulation, censorship, or the disruption of legitimate nodes' operations.
3. **Churn and Node Failures**: Chord is designed to handle nodes joining and leaving the network gracefully, but this feature can be exploited by attackers to disrupt the network's stability. Malicious nodes can artificially generate a high rate of node churn or fail abruptly to create instability.
4. **Routing Attacks**: Chord relies on a structured routing table to find nodes in the network efficiently. An attacker may manipulate this routing table to redirect requests or disrupt the network's normal operation, causing data loss or unauthorized access.
5. **Partitioning Attacks**: Chord is vulnerable to partitioning attacks, where an attacker can isolate a subset of the network from the rest. This can be used to deny access to data or manipulate data within the isolated partition without detection.
6. **Incentive Mechanisms**: In Chord-based P2P networks, there may be issues with incentivizing nodes to participate fairly. Nodes may not always behave honestly, impacting the overall integrity and security of the network.
7. **Bootstrapping Security**: When a new node joins the Chord network, it needs to locate an existing node to obtain information about the network's structure. If an attacker controls the bootstrap node, they can manipulate the joining node's view of the network or launch attacks against it.
8. **Secure Identifier Assignment**: Chord uses identifiers to assign nodes to points in the ring-based network. Ensuring secure and unique identifier assignment is crucial to prevent attacks related to node impersonation and data tampering.

To address these security issues, Chord-based applications often implement additional security measures, such as authentication mechanisms, data encryption, and reputation systems. Additionally, research is ongoing to develop more secure DHT protocols and techniques for mitigating these vulnerabilities.

It's important to note that while Chord has security challenges, it also has many merits, such as its scalability and efficiency, making it a popular choice for building decentralized systems. However, understanding and addressing its security issues is essential when implementing Chord in practical applications.

*Sakshi Shashikant Yeole (20CS01047)*