

- Presented By:
- Himanshu Pandey
- Prarabdh Garg
- Unmesh Roy

ThinQ

Today's Presentation



The Problem



A brief description of the framework



Overview of Concepts Used



Features of the Framework



Using the Framework to build applications



Challenges faced



Scope of Improvement



The Problem

- Sectors moving towards aggregation
- Leads to existence of centralized middlemen
- Middlemen do nothing more than gather data from both parties and act as a bridge
- Often leads to monopoly of middlemen, due to the existence of a very few of them, and because of the data they have collected
- Puts a huge risk to the data privacy of the users
- Higher prices for Consumers, Lower profits for Service Providers



Brief Description of the framework



A DECENTRALIZED
FRAMEWORK THAT
CONNECTS CONSUMERS
DIRECTLY TO SERVICE
PROVIDERS



CENTRALIZATION IS
PREVENTED BY USING A
DISTRIBUTED FILE SYSTEM



DATA PRIVACY IS PROTECTED
BY ENSURING DATA DOES
NOT LEAVE YOUR DEVICE



REAL-TIME
COMMUNICATION
BETWEEN PEERS IS DONE
THROUGH PUB/SUB



EVERYTHING IS TREATED AS
A FILE, INSPIRED BY THE
DESIGN OF LINUX



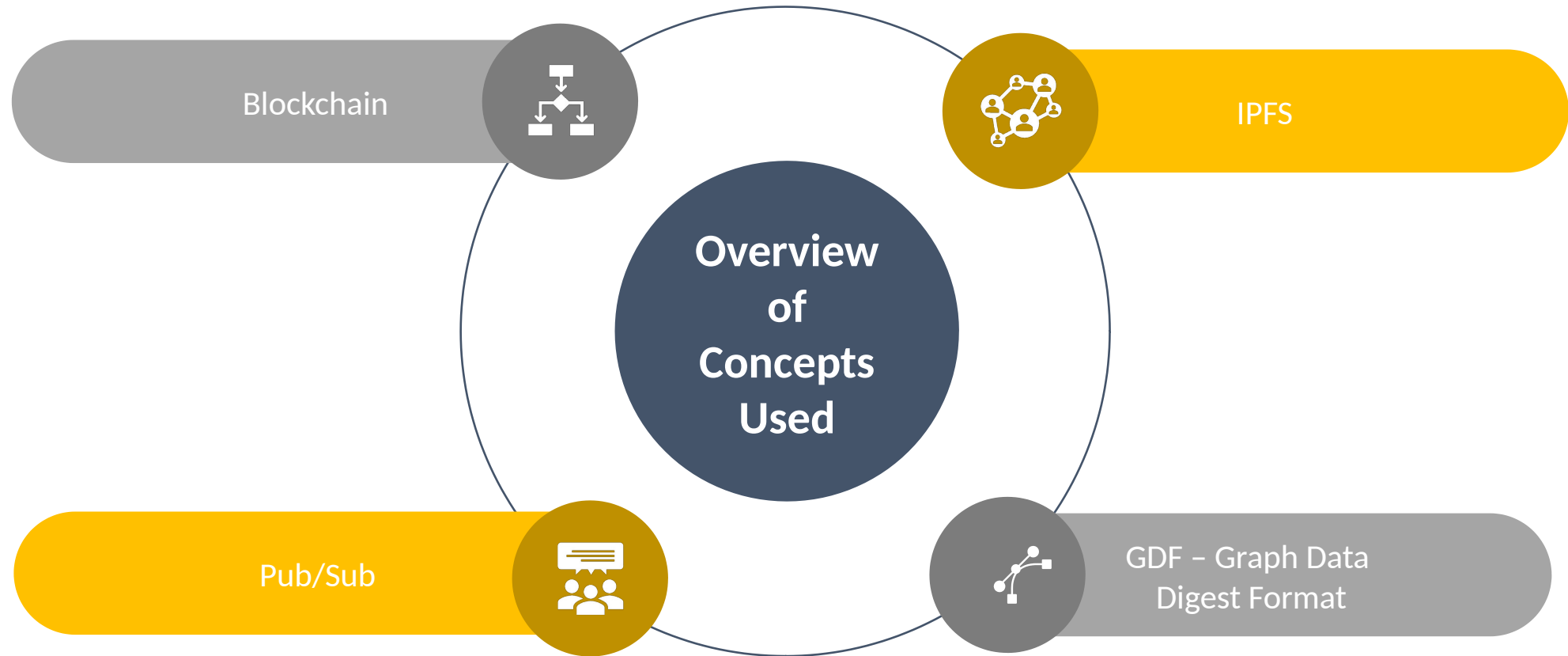
DATA IS STORED IN A GRAPH
BASED FORMAT



PRIORITIZATION OF
REQUESTS BASED ON
BEHAVIOUR ON THE IPFS
NETWORK

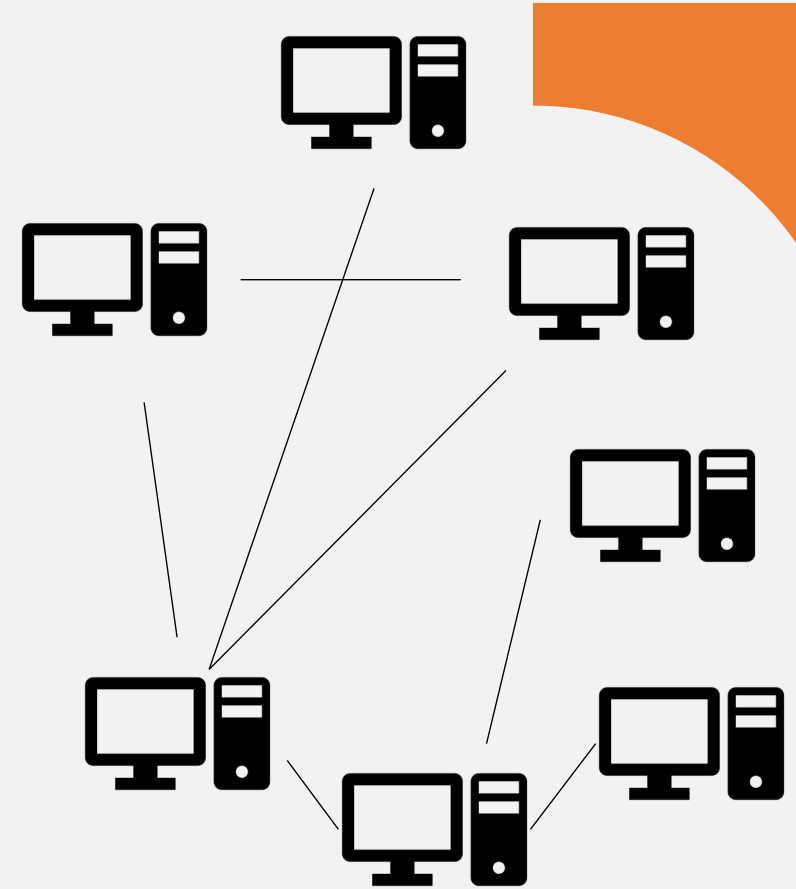


PREVENTS MONOPOLY BY
ENSURING NO CENTRAL
PARTY IN THE TRANSACTION



IPFS

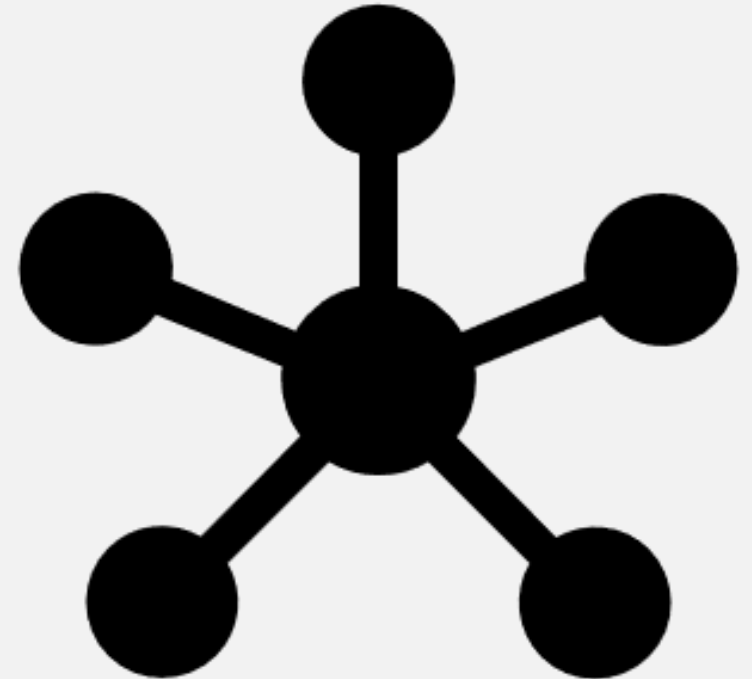
- IPFS is a distributed file system
- IPFS forms the backbone of our framework
- It uses content hashing to store and identify files
- Also uses the concept of Distributed Hash Tables to ensure the fact that any file can be accessed only through its hash
- It uses a DAG to represent files, and incentivised block exchange to transfer files between users



GDF

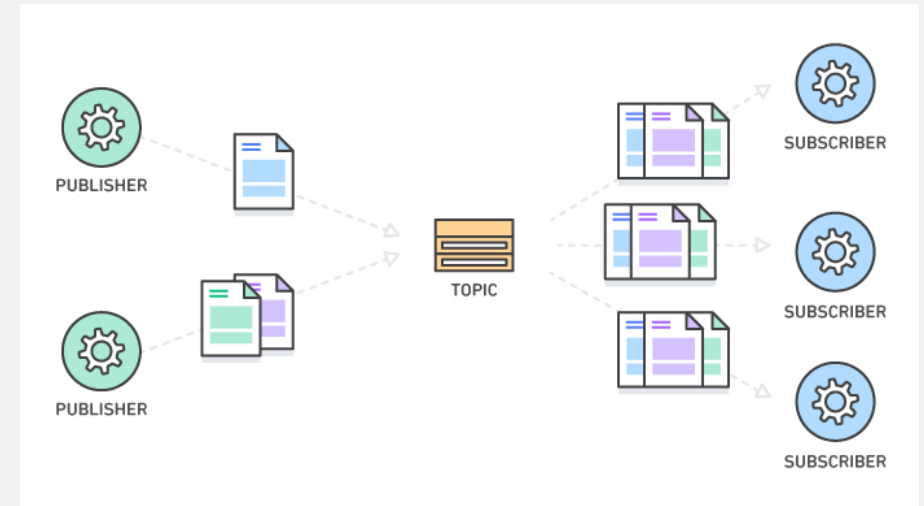
UID | Subject | Subject_Qualifier | Predicate | Predicate_Qualifier | Object | Object_Qualifier

- GDF is a Graph based data digest format which can be used to represent any form of data
- GDF is based on the graph data model which represents every relation using three entities - subject, predicate and object.
- It is a seven column fixed format, which makes it easy to store
- The project uses GDF to store and transfer messages at all points



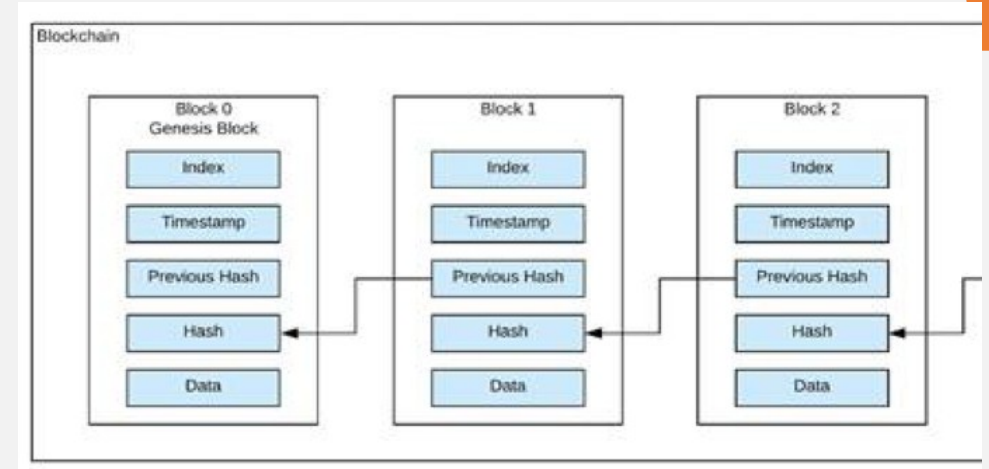
Pub/Sub

- Pub/Sub, shorthand for publish/subscribe messaging
- Asynchronous messaging service
- Decouples services that produce events from services that process events.
- publisher creates and sends messages to the *topic*.
- Subscribers then create a *subscription* to a topic to receive messages from it.
- Transfer of messages may be one-many/many-one/many-many fashion.



Blockchain

- Used to store an immutable ledger in distributed systems
- Necessary as we may require to store some internal information about users
- Can be used to store ratings and Trust points (Discussed later) in a secure way
- Can also be used to maintain a record of all the transactions that took place in a distributed manner





Features of the Framework

Address Book



- Every user has a unique IPFS id
- The id is hard to remember, so a mapping is made
- Similar to the list of contacts in your phone
- Helps establish initial trust between users
- Helps peers discover each other on the network
- Since IPFS id is just a long piece of text, it can be shared in multiple ways

QmUqUDPRH2UkdcdraMBXymqmBuXEZqa5dteqCMhvXpphu5



Real-Time Chatting

- Text messages exchanged between users in real-time
- Messages are converted to files in the IPFS file system
- Hash of file is shared
- Prevents data duplication
- Pub/Sub mechanism used
- Peers who are not online at the time of sending the message are delivered the message as soon as they come online

File Book and File Sharing

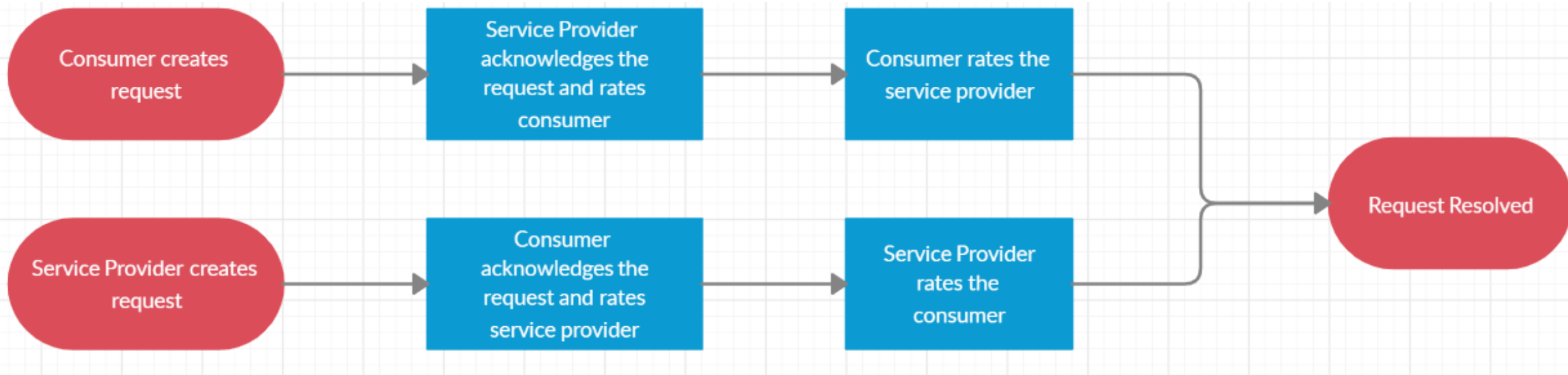
- Upload and store files on the IPFS file system
- These files can later be shared with other users on the network
- While sharing files, only the hash of the files are shared
- Same version of the file is viewed by everyone
- The file never leaves your system
- Option to pin the file, so that it is available even if you are offline

Service Requests

- Similar to messages, with the difference being that they are computer generated
- Meta-data can be added to requests, based on the application being developed
- Services need to be closed, in order to ensure that transactions are completed
- Closing of services is a three way procedure, to ensure both parties have an equal say
- Ratings are given while closing requests, which help in prioritizing requests



Process of closing requests

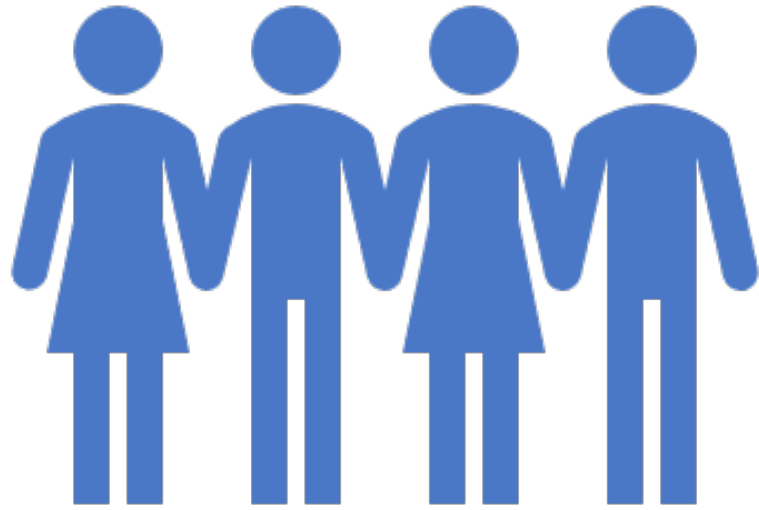




Ratings

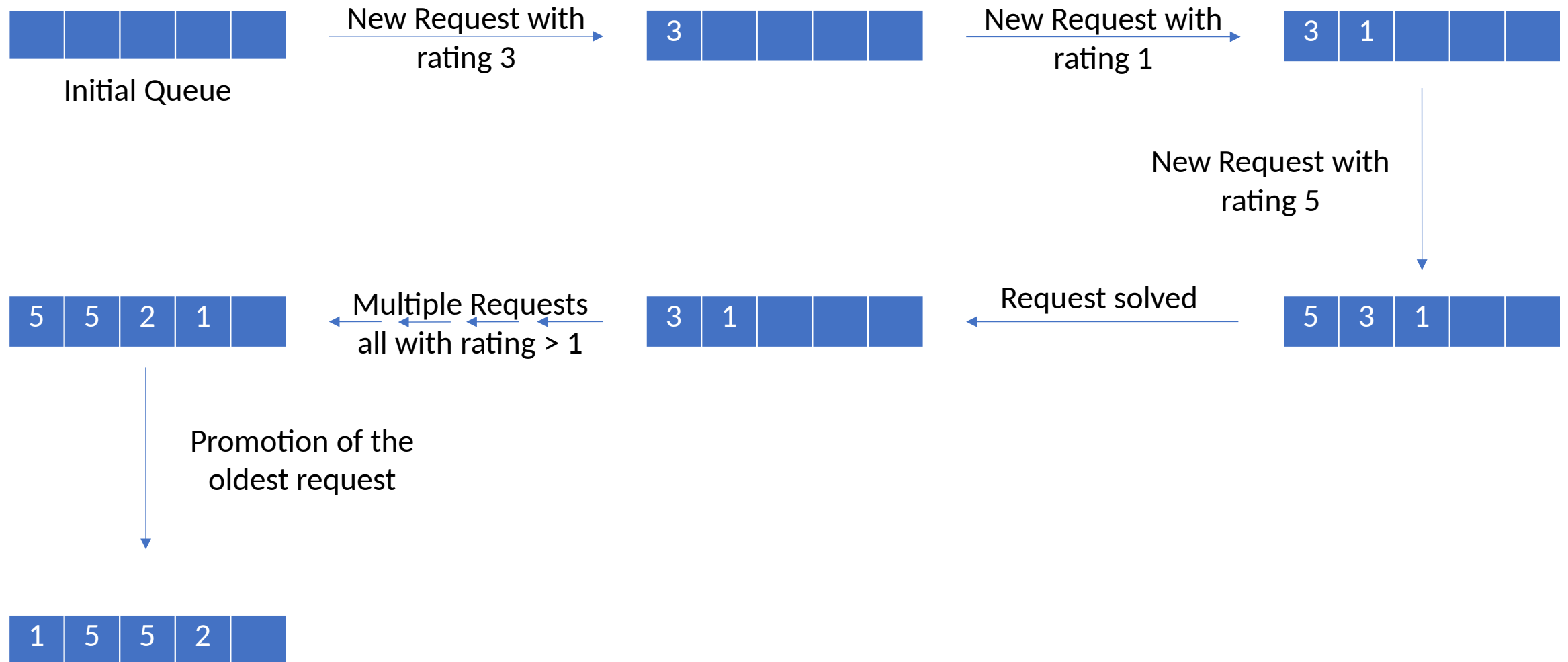
- Users rate each other after transactions on a scale of 1 to 5
- Ratings help us prioritize requests of consumers
- Also helps consumers select from multiple Service providers
- Prevents nodes from behaving maliciously on the network by providing incentives
- Currently stored in the IPFS file system of the user itself. Improvement suggested later

Request Prioritization



- Service Providers generally have limited capacity
- Introduced a concept of Social Trust Points (STP)
- Currently STP is solely determined by a users rating
- STP as well as timestamp are taken into account for deciding propriety
- Timestamps are necessary to prevent starvation

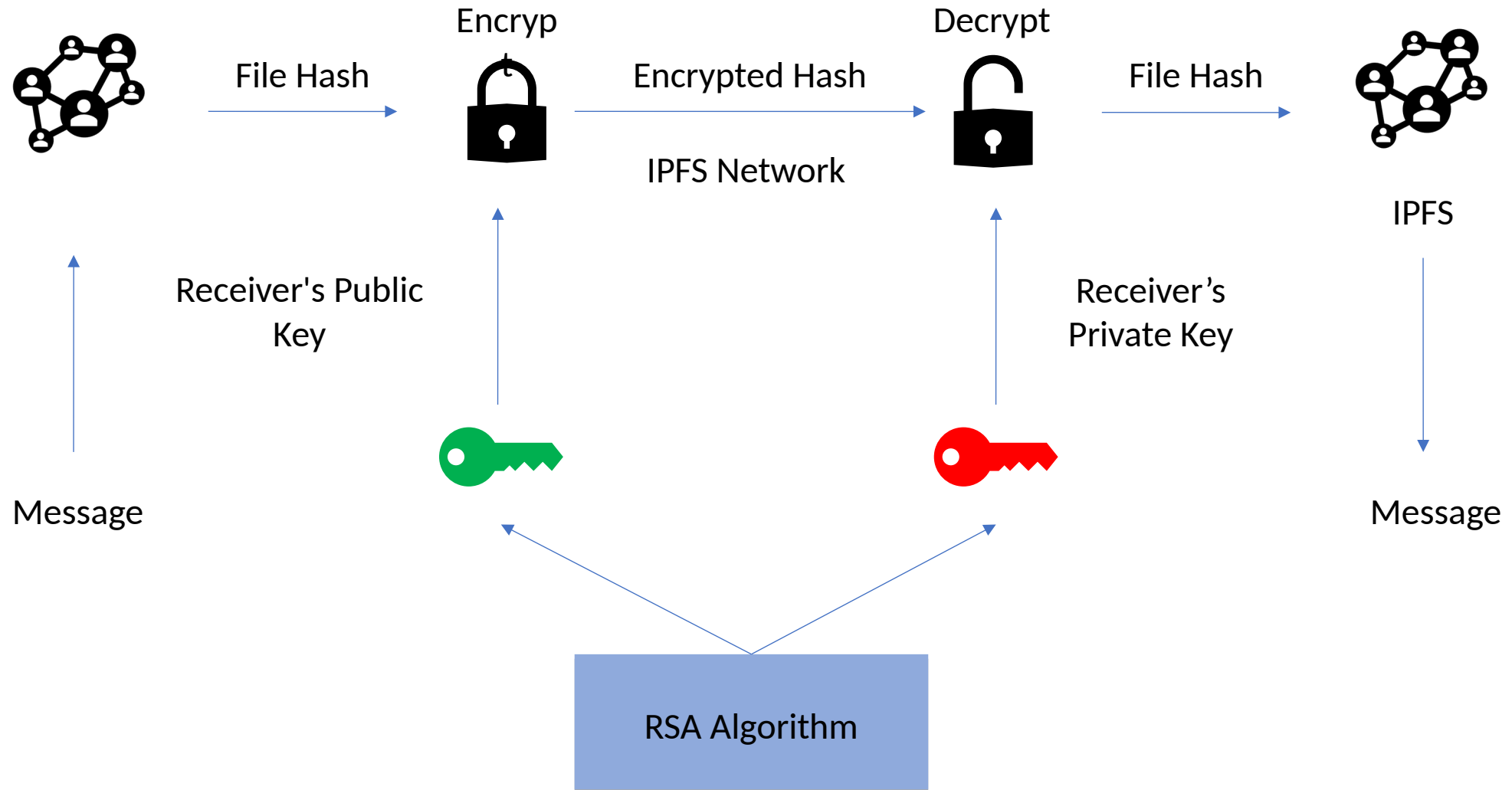
Example of how prioritization works




Encryption

- Messages are passed through Pub/Sub mechanism, hence encryption is necessary
- Since underlying medium is an untrusted one, asymmetric key cryptography was chosen
- GPG encryption, using RSA keys is used
- Currently only messages are being encrypted, but multiple layers of encryption can be implemented in the future

Example of how Encryption works



A dark blue, irregular ink splatter shape is centered on a white background. The splatter has a rough, textured edge with some smaller droplets and splatters extending outwards. The text is centered within this blue shape.

Building apps with the framework

Demonstration of ThinQ App



Challenges Faced

IPFS is a framework currently under development and new versions are released frequently. Also no proper documentation is available

We had a lot of problems in setting up the node environment to run the app on different systems

Converting into an npm package was a bit challenging as we had to decide what all functions we should expose, and what all flexibility should be given to the developer

Scope of Improvement



The ratings that are currently stored with the user can be stored in a Blockchain to improve security



A method of improving the current STP system, based on the need of the application, so that the value of incentives can vary based on the use case



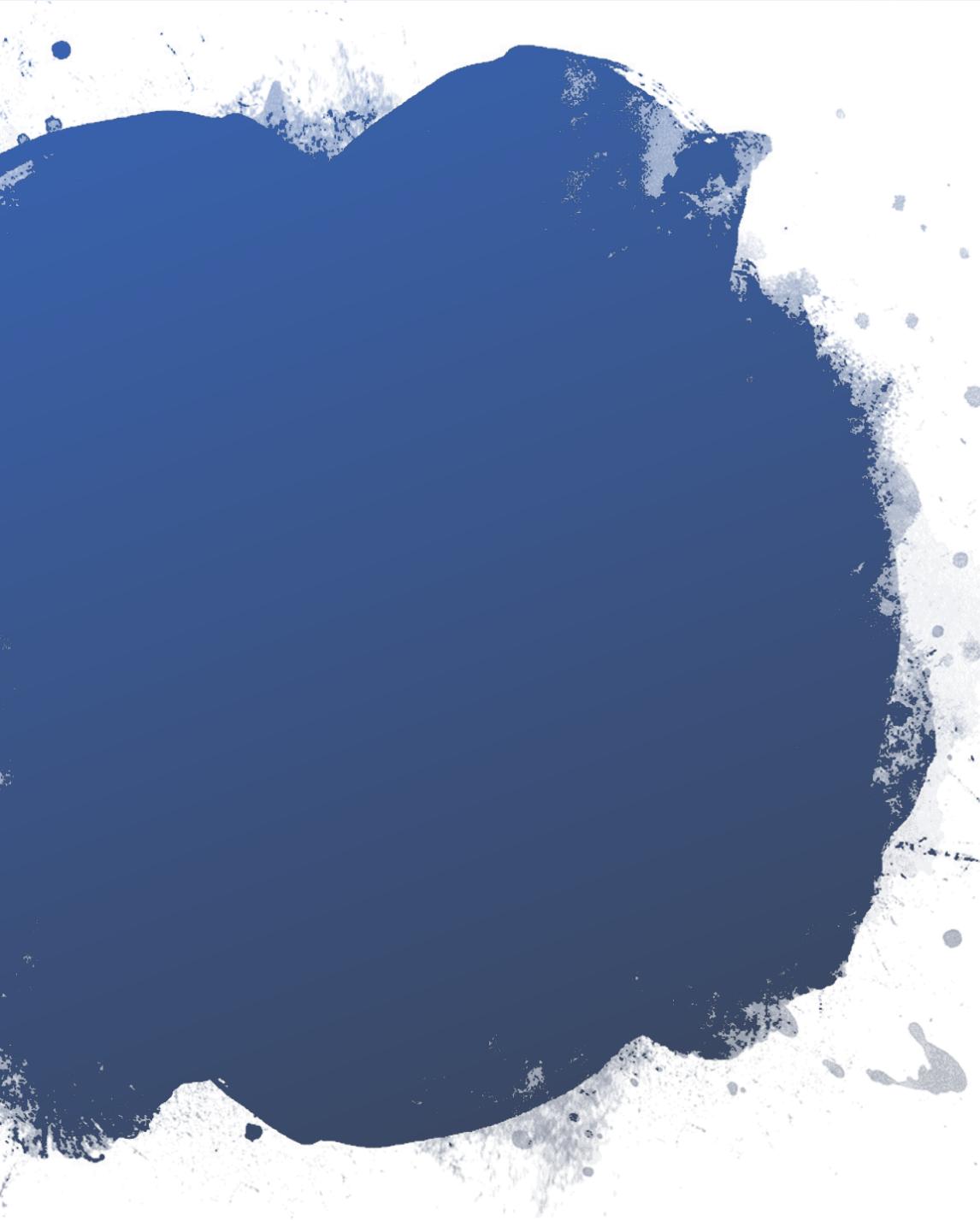
Implementation of various layers of encryption that can be chosen by the developer at the time of designing apps based on their requirements



Design and implement a model to capture the underlying social relationships between users, and make trust calculation a bit more indirect

Individual Contributions

- Himanshu Pandey
 - Address Book Handler functions (Chat app)
 - Routers (Chat app)
 - File Sharing (Chat app)
 - File book (Chat app)
 - Service Request Closing System (ThinQ)
 - Rating System (ThinQ)
- Prarabdh Garg
 - IPFS node setup (Chat App)
 - SQLite3 local database setup (Chat App)
 - FIFO Queue Management (Chat App)
 - Social Trust Points (ThinQ)
 - Prioritizing Service Requests (ThinQ)
 - Encryption (ThinQ and Chat app)



- Unmesh Roy

- GDF encoding functions (Chat App and ThinQ)
- Web Sockets (Chat App)
- IPFS and Database setup (ThinQ)
- User Interface (ThinQ)
- Updating Information and bio (ThinQ)
- Sending and Receiving Requests (ThinQ)

Important Links

- Framework Repository (
https://github.com/PrarabdhGarg/thinq_lib)
- Example Apps Repository (
<https://github.com/RUnmesh/ThinQ-Examples>)
- Chat Application Repository (
<https://github.com/PrarabdhGarg/ThinQ/tree/trial>)





Thank You