INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 5, Issue 2)
Available online at: www.ijariit.com

Decentralized file sharing application in a global approach

K. Mohan Krishna
kmohankrishna.15.cse@anits.edu.in
Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, Andhra Pradesh

M. Kranthi Kiran
mkranthikiran.cse@anits.edu.in
Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, Andhra Pradesh

P. Kiran Kumar
pulaparthikiran@gmail.com
Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, Andhra Pradesh

M. Tejasmika
tejasmika88@gmail.com
Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, Andhra Pradesh

K. Ravi Teja Reddy
ravitejareddy321@gmail.com
Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, Andhra Pradesh

K. Sandeep
sandeepchowdary@gmail.com
Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, Andhra Pradesh

ABSTRACT

Use of semantic technologies has been prevalent in sharing files and media over the internet. This project proposes a new cooperative file sharing and storage system that allows users to store, share and find files over a robust and scalable network. This system is based on peer-to-peer technologies and provides a self-managed, decentralized, efficient and distributed way to store and retrieve files on peer nodes. Each node donates storage space to the system. The system also provides its users with a global hierarchical directory structure to browse for files like a file-system maintaining RDF Graphs forming Ontology in individual systems with the implementation of peer to peer communication in global scenario overcoming the drawbacks of Network Address Translation using TCP Hole Punching technique.

Keywords— *Resource description framework, TCP hole punching, Decentralized*

1. INTRODUCTION

As the world is adjusting to the conventional application, the whole ecosystems are also evolving. Decentralized applications are a type of applications which are not owned by the single authority, cannot be shut down or cannot have downtime. The concept of the decentralized application is still in its nascent state and are gaining popularity day by day. These decentralized need not be running on a blockchain mechanism , they may have their own methods and protocols of transferring content, an example of such is bit torrent.

Also due to an increase in the number of digital research articles, there is a need for the computer science research community to efficiently organize and retrieve these files. Many digital libraries available online which own many research articles. Our Semantic Application helps the research community to have ownership of their research articles and enable them to share among the community connected through our network. This eliminates the concept of ownership of research articles by a particular third party.

The above mentioned decentralized application aims at establishing a network of computers all over the globe, which can share or transfer files among themselves. No single computer in the network has the entire authority over the application, files are stored in the individual machines and are shared among the peers upon request from a node in the network.

Every single system connected in the network is considered as a node and is allocated a storage folder in order to maintain the articles belonging to the particular node. Every node behaves as a client when requesting a file and also it acts as a server when needs to send a file to another node which has requested for the particular article. Every node creates an XML format file for every individual article belonging to that particular node and builds an ontology using an RDF graph [1].

All the nodes get connected to the publically accessible server with a static IP in order to get connected to the network. Every node is provided with an interface to search for articles, the user can search by giving a title or author name of the article. When a particular node requests for an article, it is sent to the server and request is broadcasted to the other nodes in the network. The other nodes receive the request and process the request and replied to the server. When the article requested is found to be existing with a particular node with the server initiates hole punching mechanism [2] to establish a direct connection between the nodes. When a successful connection is established a direct file transfer is initiated between the nodes. Thus the articles are shared directly by the user without uploading them without any third party.

2. LITERATURE SURVEY

Users when searching for research articles opt to search through giving the title name or author name of a particular research article. As we are developing decentralized file sharing application for sharing research articles we studied architecture of Semantic Web, IPFS (InterPlanetary File system)[3] which stores data in the same file structure known as Merkle DAG which combines Merkle Trees (used in block chains), DAG (Used in Git Version control). When a file is added to the IPFS a unique has his generated. Using the Unique hash value the file can be retrieved at any node in the IPFS network. Based on this reference we designed our application to store s the Metadata of every article in form of SPO (Subject, Predicate, Object) structure and these SPO objects are combined to form a graph which termed as ontology using RDF methodology.

Our application is implemented in the global scenario, hence it faces issues with the routers (Network Address Translation). Network Address Translation (NAT) causes well-known difficulties for peer-to-peer (P2P) communication, since the peers involved may not be reachable at any globally valid IP address. The issue with NAT is that remote computers cannot initiate sends to local computers because no mapping yet exists. Therefore, if two computers both behind NAT try to connect, neither will be able to do so. This is a problem with voice communication, peer to peer games, and many another peer to peer applications where your users host and the host is behind a NAT. Users would have to go to their router configuration screen and set up a mapping. But it is impossible to setup up such settings at every router when the application is being implemented under a global scenario.

Hole punching (or sometimes punch-through) [2] is a technique in computer networking for establishing a direct connection between two parties in which one or both are behind firewalls or behind routers that use network address translation (NAT). To punch a hole, each client connects to an unrestricted third-party server that temporarily stores external and internal address and port information for each client. The server then relays each client's information to the other, and using that information each client tries to establish a direct connection; as a result of the connections using valid port numbers, restrictive firewalls or routers accept and forward the incoming packets on each side.

When a file is to be transferred directly between the nodes the hole punching module is initiated in the server to establish a direct connection between two nodes. Once the successful connection is established, a direct connection is initiated between the remote hosts.

3. ARCHITECTURE

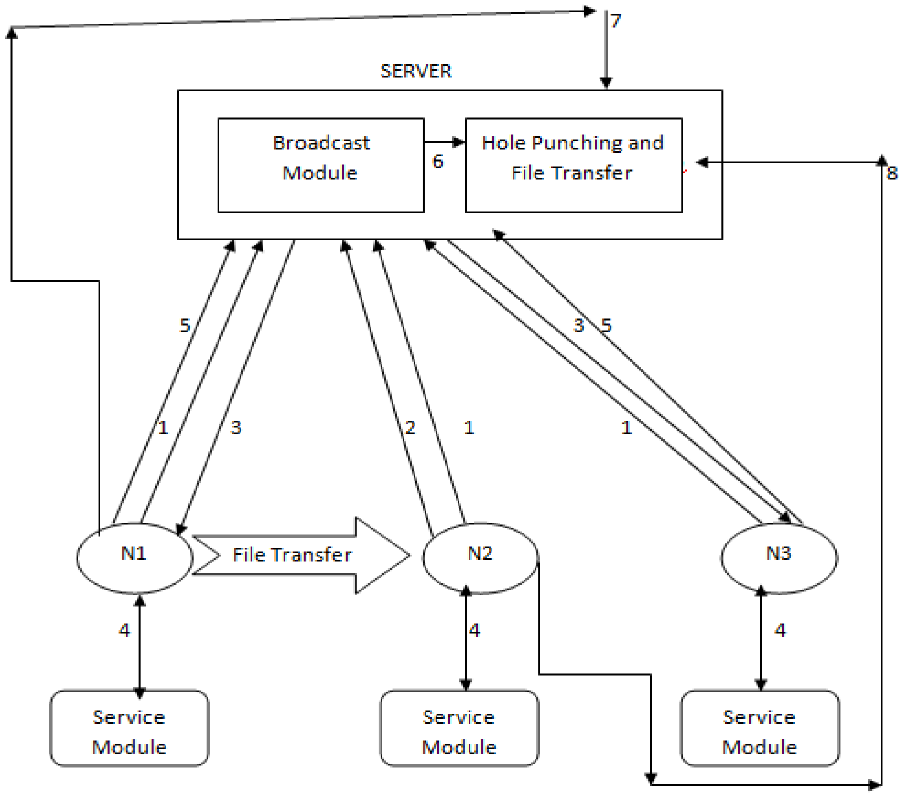


Fig. 1: Describing the functionality of the application in a global approach

Server: Server is an AWS (Amazon Web Services) windows machine with a publically static IP address. The server consists Broadcasting module, Hole Punching and file Transfer Module.

Broadcasting Module: The broadcasting module receives a request from the node and broadcast to remaining nodes in the network.

Hole Punching and File Transfer Module: This module is initiated by a broadcasting module if the requested file is found. The hole punching module is used to establish a direct connection between the nodes by traversing the NAT.

1. All the nodes get connected to the broadcasting module in the server machine.
2. A node sends a request for a particular article.
3. The request is received by the broadcasting module and is broadcasted to remaining nodes in the network.
4. Each node receives the network and sends to service modules which reply whether the requested file is existing within the system or not.
5. The status is sent back to the broadcasting module.
6. If the broadcasting module finds that the requested file is existing with any of the nodes, it initiates hole punching module between both nodes.
7. Both the nodes send a dummy packet to the hole punching module for the process of hole punching and direct connection is established between both the nodes.
8. Direct file transfer is initiated between both the nodes and file is received in the requested node.

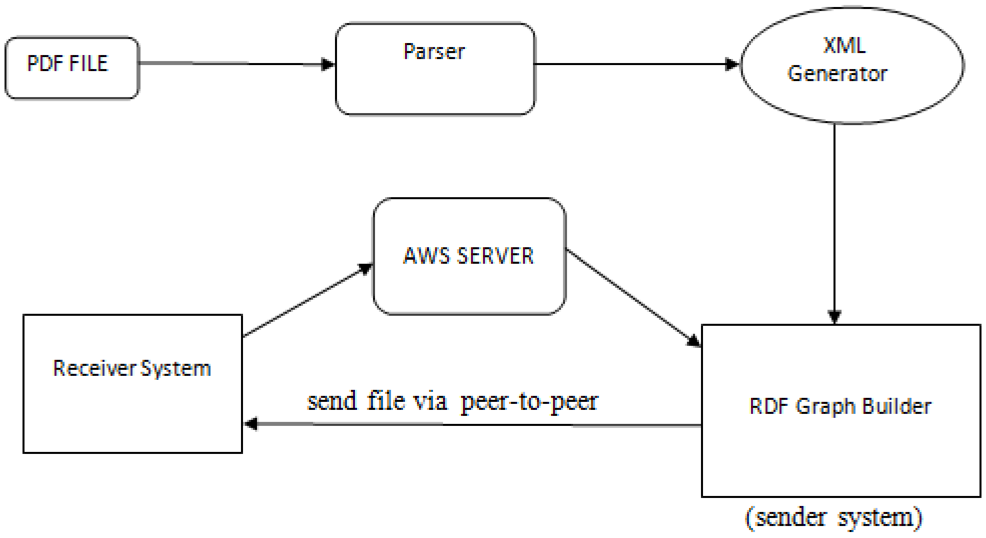
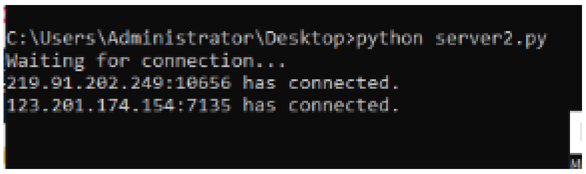


Fig. 2: Describing the functionality of a service module for an individual node

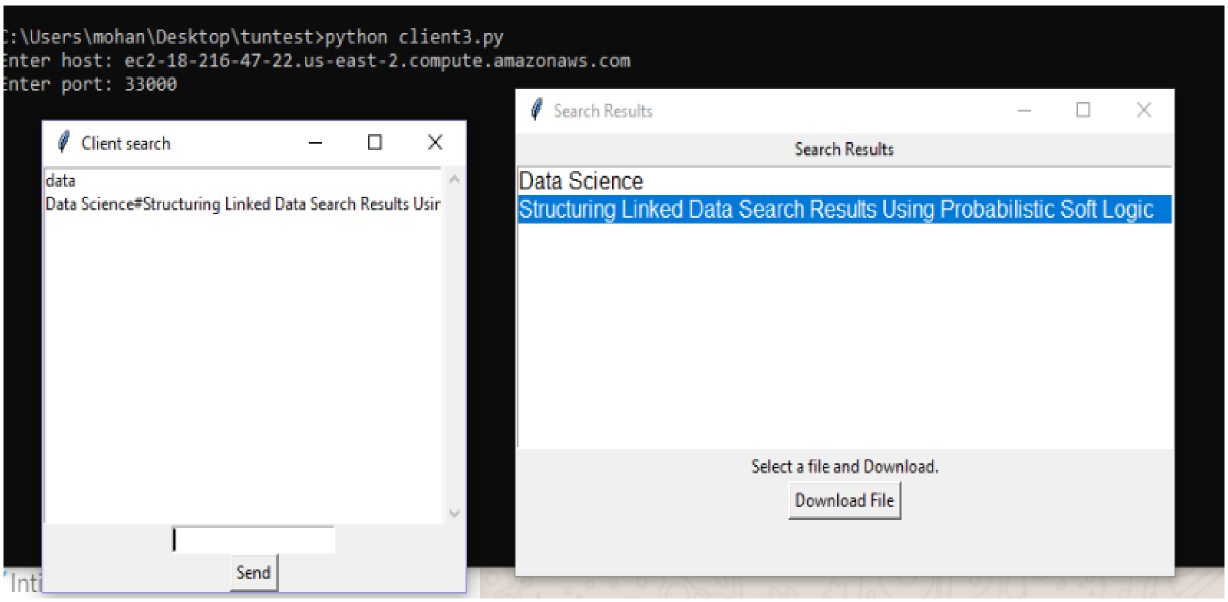
Parser: Parser is used to extract the title and author name for a particular article.
XML Generator: This module is used to generate the XML files of extracted metadata of each individual article in the system.
RDF Graph Builder: This module will take all the XML's from the above module and generate a graph which is in SPO terminology.

4. RESULTS



```
C:\Users\Administrator\Desktop>python server2.py
Waiting for connection...
219.91.202.249:10656 has connected.
123.201.174.154:7135 has connected.
```

Fig. 3: Broadcasting module listening to incoming connections



```
:\Users\mohan\Desktop\tuntest>python client3.py
nter host: ec2-18-216-47-22.us-east-2.compute.amazonaws.com
nter port: 33000
```

Client search

data
Data Science#Structuring Linked Data Search Results Usir

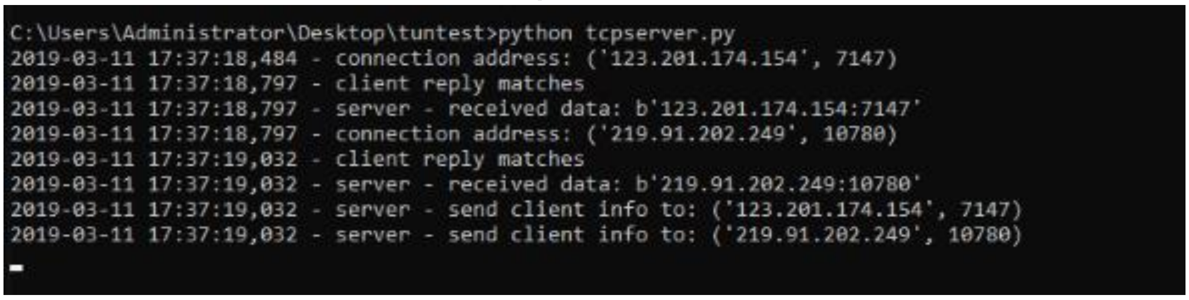
Search Results

Search Results

Data Science
Structuring Linked Data Search Results Using Probabilistic Soft Logic

Select a file and Download.

Fig. 4: User Interface for searching and downloading file



```
C:\Users\Administrator\Desktop\tuntest>python tcpserver.py
2019-03-11 17:37:18,484 - connection address: ('123.201.174.154', 7147)
2019-03-11 17:37:18,797 - client reply matches
2019-03-11 17:37:18,797 - server - received data: b'123.201.174.154:7147'
2019-03-11 17:37:18,797 - connection address: ('219.91.202.249', 10780)
2019-03-11 17:37:19,032 - client reply matches
2019-03-11 17:37:19,032 - server - received data: b'219.91.202.249:10780'
2019-03-11 17:37:19,032 - server - send client info to: ('123.201.174.154', 7147)
2019-03-11 17:37:19,032 - server - send client info to: ('219.91.202.249', 10780)
```

Fig. 5: Hole punching in the server

5. CONCLUSION

The above proposed decentralized application solves the issue of ownership of articles of research articles by a third party, by allowing the users to get connected to a common network and search, share, download research articles with other users.

6. REFERENCES

[1] Matching RDF Graphs, Jeremy J. Carroll link.springer.com/content/pdf/10.1007%2F3-540-48005-6_3.pdf

[2] Peer-to-Peer Communication Across Network Address Translators, Bryan Ford https://bford.info/pub/net/p2pnat/

[3] Inter-Planetary file system https://docs.ipfs.io/