



Supervisor: Frank Jiang
Assistant Supervisor: Allan NG

SIT723 – Research Project A

Report on Peer-to-Peer model
evaluation for protection against
Advanced Persistent Threat (APT)

TEJA KUMAR REDDY KOVVURI
ID: 219411066

Table of Contents

Introduction and Motivation	2
Literature Review	2
Existing Work	7
Research design	8
Client-Server Model:	8
Roadmap:	9
Peer-to-Peer Model:	10
Roadmap:	11
Artefact Development:	12
Experiment on Client - Server model	13
The Procedure for the experiment	13
Experiment on Peer – to - Peer model	15
The Procedure for the experiment	16
Evaluation	18
Summary Table for comparison of Client-Server model and Peer-to-Peer model:	18
Future Work	20
References	20
Appendix	22

Introduction and Motivation

Cyber-attack has been increasing rapidly. Most of the victims are targeted by the hacker using APT (Advanced Persistent Threat). This is a long and sophisticated attack process which required knowledge of the victim, type of tool and most importantly skill.

In this project, we are not going to investigate the entire APT attack. We will focus on the initial phase of the attack called recon. This is the first phase before any attack. Tool like nmap, tcpdump and wireshark can be used to collect traffic in the network. Even though, encryption is implemented, this is inadequate to keep the determined hacker from retrieving information. Information such as source & destination ip address, source & destination port and application version can still be retrieved from the traffic packet. This information gives enough information about the victim's system.

Literature Review

An Advanced Persistent Threat (APT) is defined as a sophisticated attack consisting of several steps carried out by the experienced and highly skilled actors with a great determination to achieve their goal (Brahim ID Messaoud, 2016). Most organizations have the primary security concern about the Advanced Persistent Threat (APT) towards their technical infrastructure which have the potential impact of serious damage to organizations from cybercriminals (Dev Bhatnagar, 2019). In fact, there is a significant risk to every organization due to the Advanced Persistent Threats (APT) because of the sophistication which allows attackers to bypass security systems and largely infiltrate the target network (Martin Ussath, 2016). The attack and the defence system to counter APTs must involve advanced planning and strategies like military operations. The current cyber-security-methodologies are not sufficient to adequately address APTs (Sun et al., 2020). In this Advanced Persistent Threat, the initial phase known as Recon is used to gather information about the target like employees, organization's IT infrastructure, assets, network and many more (Alshamrani et al., 2019). As communication is also an essential part and that consists of sensitive information, adversaries target these for meta-data about sender and receiver and other sensitive information. The traditional way of communication used to happen through a centralized node and that was exploited more often by attackers (Zhang et al., 2020).

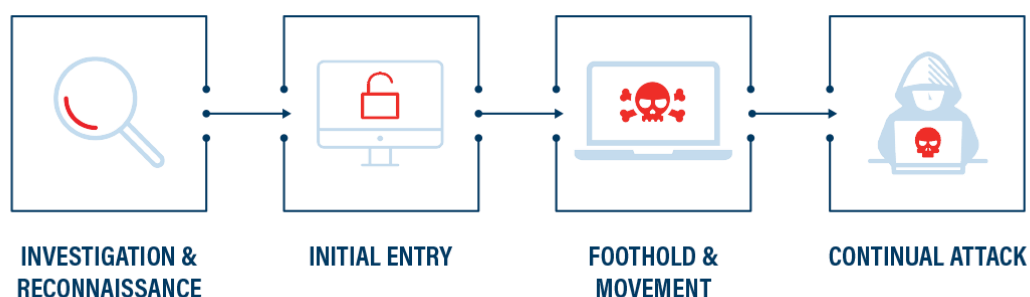


Figure 1-The phases of an advanced persistent threat (APT) (TEAM, 2019)

When it comes to sharing data with multiple clients, a Client-Server model has been used in the earlier days. This model has been used every day for different applications. Different functions will be carried out by the server and client during the exchange of data with the provision of inter-process communication (Oluwatosin, 2014). The scalability issues in point-to-point model have been addressed with the help of the Client Server Model. A special server will serve the requests of all the clients in the network. It is also called “many-to-one” architecture (RTI, 2015).

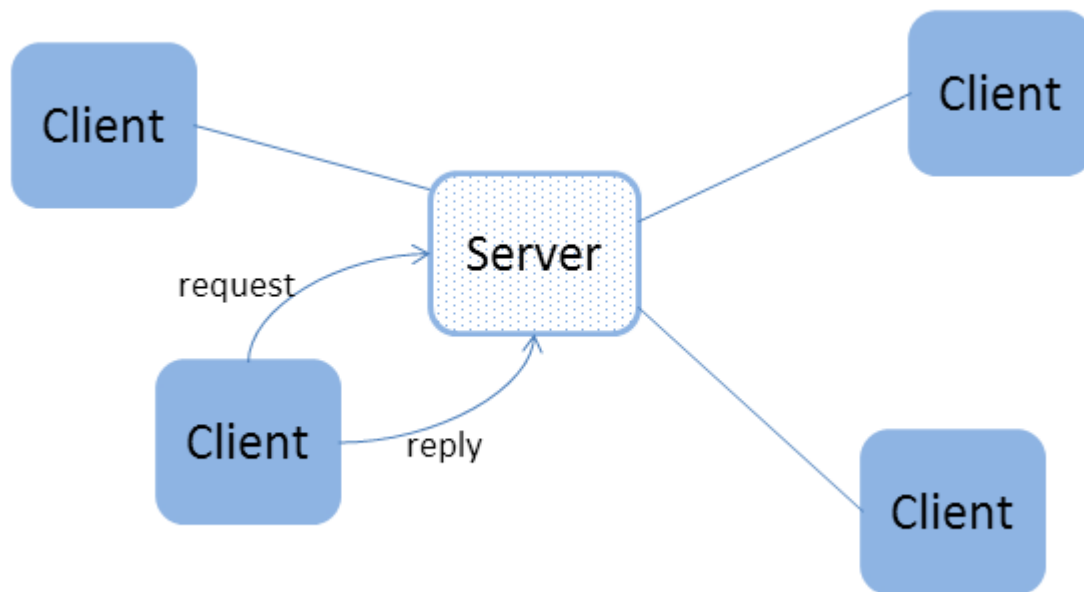


Figure 2-Client-server is many-to-one communications (RTI, 2015)

Several drawbacks have been identified in this model which are Security, Interoperability, reusability, and performance. The security aspect raised concern with the model (Sawsan Ali Hamid, 2020). The client server model reveals the actual source and destination addresses to any adversaries during data transmission (Duchessi and Chengalur-Smith, 1998). Due to this, several attacks have been happened.

A peer-to-peer model has been used to overcome the drawbacks of Client Server Model. Any computer can act as a server or a client depending on what is most appropriate to the system at the time (Loo, 2003). There is no central coordination by servers and every node communicates with the other nodes in the network and they mostly exchange files (S Venu Gopal, 2016). The security in Peer-to-Peer Model satisfies the Confidentiality, Integrity, and Availability (CIA) triage (Stefan Kraxberger, 2009). The cost for implementing the Peer-to-Peer model is less compared to Client Server Model. Some other advantages of Peer-to-Peer model are Reliability, Scalability, and sharing of resources among all the nodes equally (Roomi, 2020, November 15).

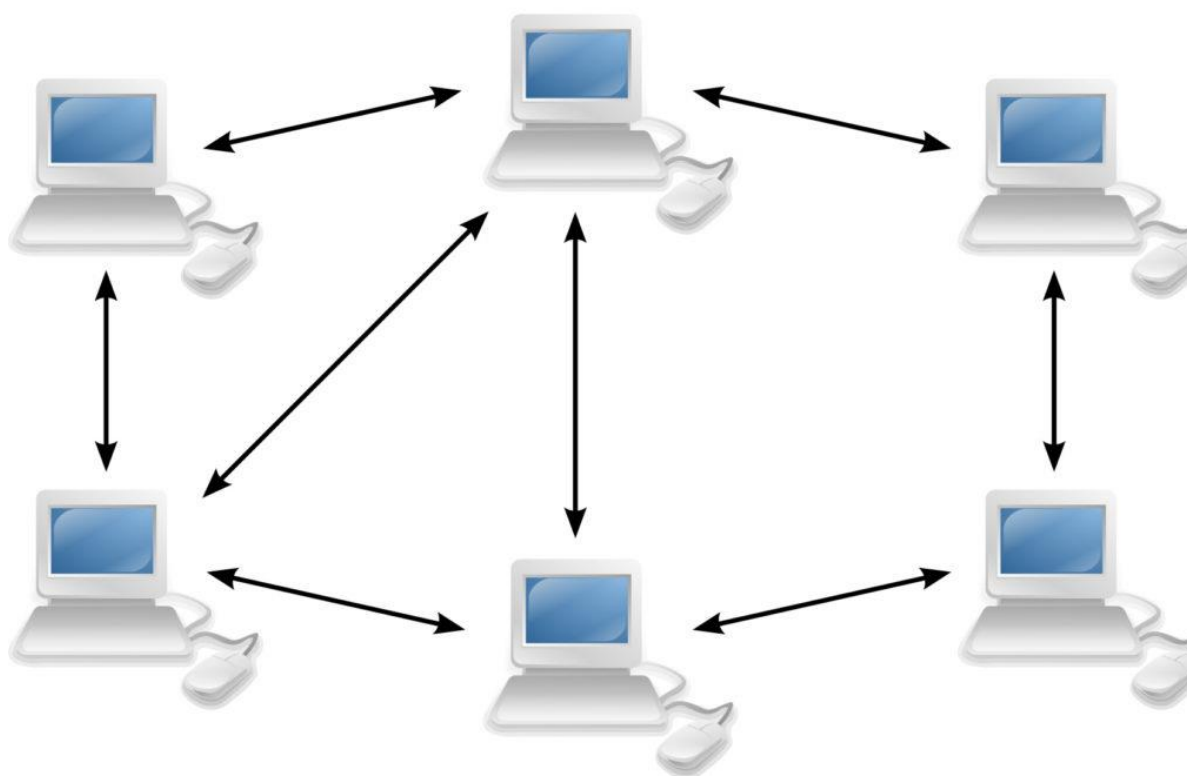


Figure 3-Peer-to-Peer Networks (CyberAgents, 2017)

IPFS (Inter-Planetary File System) protocol has been used for implementing the Peer-to-Peer model. It is a decentralized system based on a peer-to-peer protocol. It stores each file with a unique fingerprint called a cryptographic hash according to the content of the file (Huang et al., 2020). All the nodes are interconnected in the network and the data will be exchanged between the nodes. Adversaries cannot find the details about data from which source to which destination the data has been transferred.

IPFS stores and retrieves IPFS objects for Pee-to-Peer system. IPFS object is a data structure with two fields:

Field Name	Description
Data	It is an unstructured binary data blob with size <256 KB.
Links	These are array of Link structures. These are the links to other IPFS objects.

Table 1-IPFS Object data structure

Each Link Structure has three data fields:

Field Name	Description
Name	The name of the Link.
Hash	It is the hash of the linked IPFS object.

Size	It is the size of the linked IPFS object, and it consists of the size of the links following this.
-------------	--

Table 2-IPFS Object Link Structure (Infura, 2021)

Multicast DNS (MDNS) is used to establish communication between the nodes in the IPFS network (IPFS, 2021a). Whenever the communication gets started, the first communication happens between the nodes in the IPFS network is the exchange of MDNS information. This will setup the connections between the nodes and makes IPFS ready for exchanging files/data. The figure 1 in Appendix Section shows the MDNS between nodes.

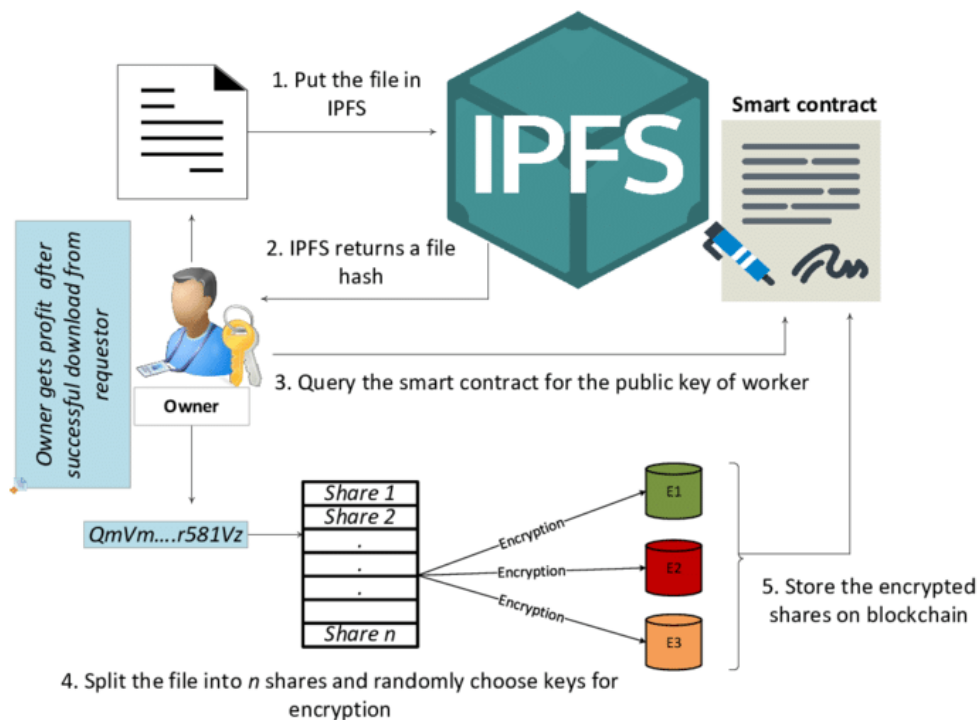


Figure 4-Data sharing on IPFS by owner (Javaid, 2019)

Another protocol which was also taken into consideration is the Whisper protocol. Whisper is a peer-2-peer decentralized messaging protocol used to maintain secrecy (Technologies, 2021). Whisper works on the off-chain part of DApps which means that there is no correlation between whisper and blockchain. In Whisper protocol, the information is stored in a payload which is then encrypted by applying the padding (Zhang et al., 2021). This protocol is used to achieve darkness which is a way of anonymity maintained between the nodes. This means the information about sender or receiver cannot be attained through packet analysis. The communication happens only on the basis of hashes of the recipients and 100% obfuscated communication is maintained (Tomu, 2018).

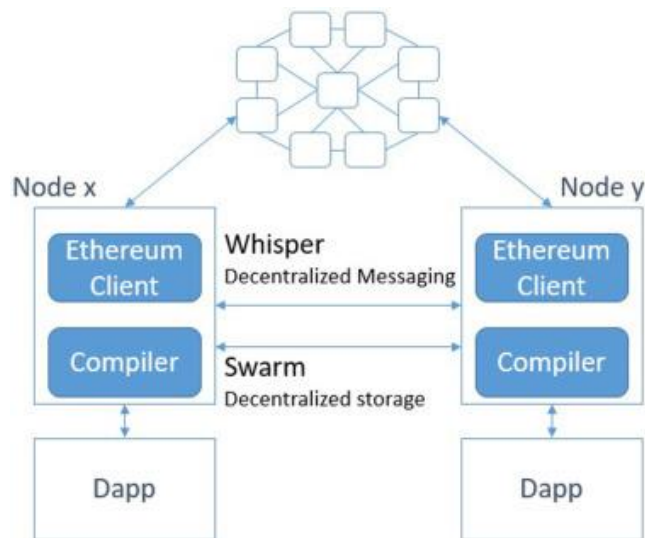


Figure 5-Whisper Protocol (Joanna Moubaraka, 2020)

Whisper nodes send and receive envelopes which are the packets. The envelop consists of encrypted payload and some metadata in plain format, because this data will be used for decryption. The envelop has the following format:

Header	Description
Version	Upto 4 bytes. It indicates encryption method. If Version is higher than current, envelope could not be decrypted, and therefore only forwarded to the peers.
Expiry time	4 bytes (UNIX time in seconds)
TTL	4 bytes (time-to-live in seconds)
Topic	4 bytes of arbitrary data
AESNonce	12 bytes of random data (only present in case of symmetric encryption)
Data	Byte array of arbitrary size (contains encrypted message).
EnvNonce	8 bytes of arbitrary data (used for PoW calculation).

Table 3-Whisper Envelop Format (Ethereum, 2019)

Whisper nodes know nothing about content of envelopes which they cannot decrypt. The nodes pass envelopes around regardless of their ability to decrypt the message, or their interest in it at all. This is an important component in Whisper's dark communications strategy.

Ipfs has been chosen as the protocol than Whisper protocol for this experiment to implement the Peer-to-Peer network and to transfer data.

Features	IPFS Protocol	Whisper Protocol
Usage	It is used to Used to transfer files from one node to the other nodes	It is a Messaging protocol which sends exchanges messages between the nodes
Node Security	Source and destination addresses are not visible (i.e., anonymization will be maintained)	Whisper maintains darkness for nodes (darkness helps in maintaining anonymization)
Data Security	Data is encrypted during transmission	Messages are encrypted during transmission
Address	Node is identified based on the ip address along with the type of protocol	Node is identified based on ip address
Protocol	It can be operated using two protocols: Libp2p IPFS	It can be operated using one protocol: Whisper

Table 4-Comparison of IPFS Protocol with Whisper Protocol

Existing Work

There are three fundamental principles to understanding IPFS:

1. Unique identification via content addressing
2. Content linking via directed acyclic graphs (DAGs)
3. Content discovery via distributed hash tables (DHTs)

IPFS can be setup for establishing connection with other nodes which are around the world. It has been achieved with the help of Internet. IPFS is also available as Desktop version which is called IPFS Desktop. By establishing IPFS globally, any node can send and receive files/data from other nodes which are part of the IPFS global network. This has been implemented in the existing works (IPFS, 2021b).

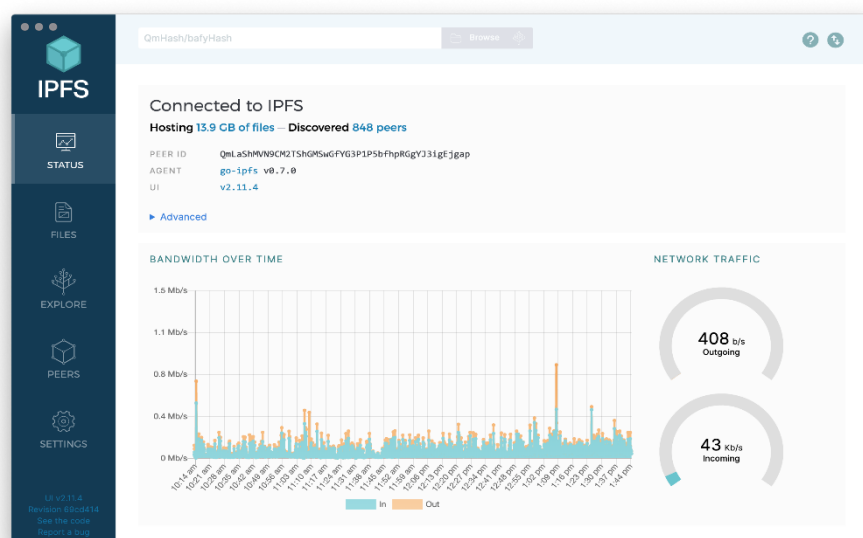


Figure 6-IPFS Desktop

In this paper, IPFS has been setup privately which is isolating the nodes in the private IPFS network from the global IPFS nodes.

Research design

Scientific Methodology has been used for this research and experiments have been conducted.

The virtualization software named “Oracle VM VirtualBox” has been used to create the virtual machines (Server and Clients) for the experiments. Along with these, the Kali Linux Virtual Machine has also been used for Wireshark to capture the traffic which is generated during data transmission and results were evaluated. The Client-Server Model (using Python Programming Language) and Peer-to-Peer Model (using GO Language) have been setup to conduct the experiments.

In addition to that, the network settings in Virtual Box for all virtual machines were set to use “Host-Only Adapter”. In this way, the traffic captured is only from these virtual machines and that gives clear results and evaluation will be quick and easy.

Along with this, the Promiscuous Mode in network settings for Kali Linux should be set to Allow All so that then Wireshark will capture all the traffic of the data transmission happened between the Server and the Clients. The Figure (1) in Appendix Section represents this setting.

Client-Server Model:

The Client-Server model is set up using python client server library. Three virtual machines are used with one of the virtual machines acting as the server. The information for the virtual machines is as of below.

Server Machine	
Name	Ubuntu ipfs Server
IP address	192.168.56.103
Client 1	
Name	Ubuntu ipfs Client – 1
IP address	192.168.56.104
Client 2	
Name	Ubuntu ipfs Client – 2
IP address	192.168.56.105

Table 5-Information of Virtual Machines

The server machine acts as the central file storage system. It stores the files and transfers file to the client when requested. The client machine requests and wait for file from the server machine.

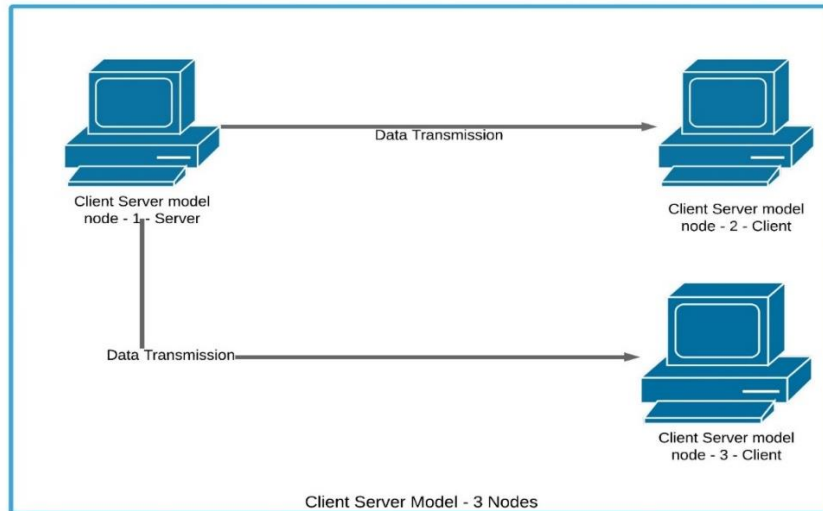


Figure 6-Client-Server Model

Roadmap:

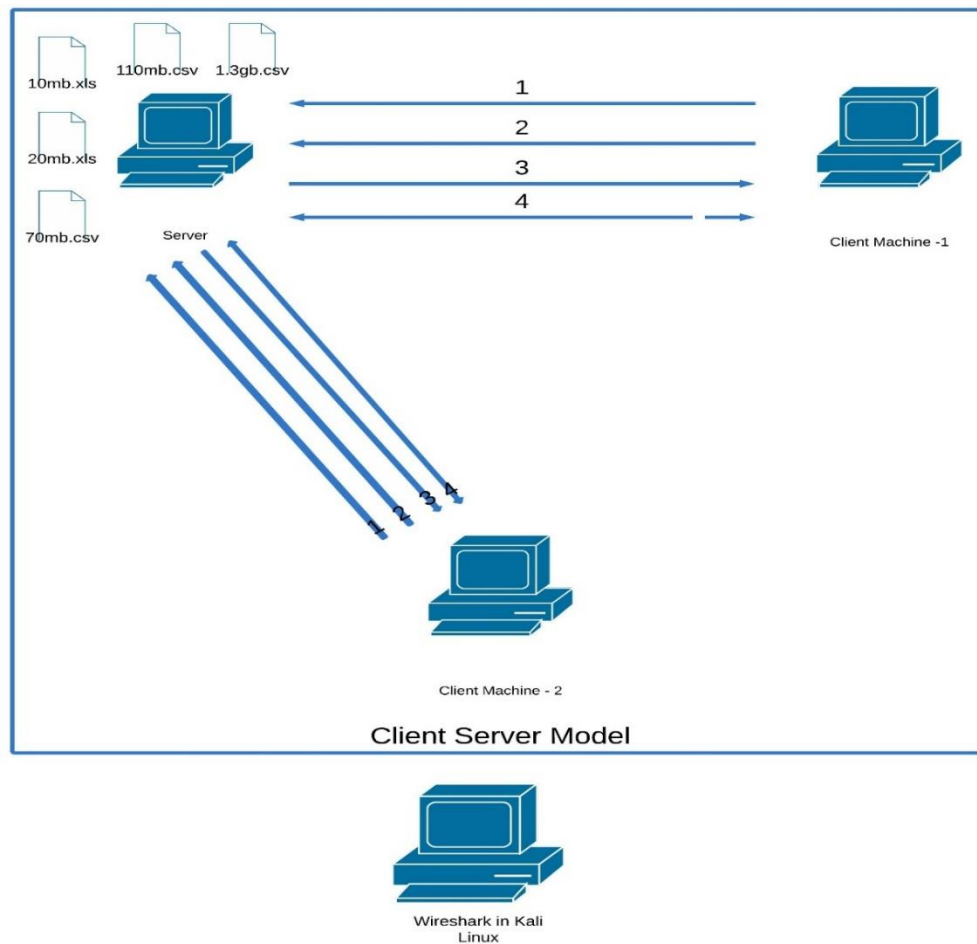


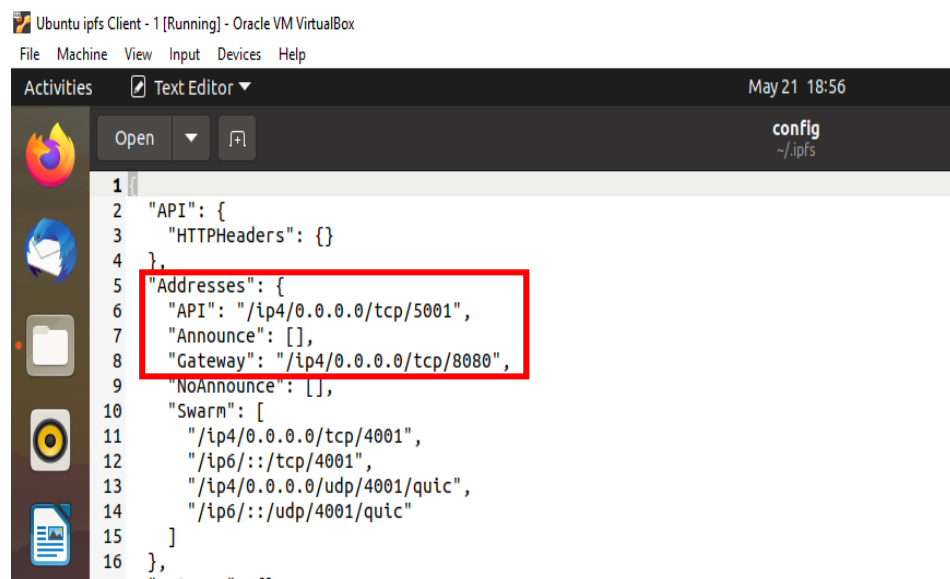
Figure 7-Roadmap of Client Server Model - 1. Client Machine 1 or 2 connects with Server Machine. 2. Client Machine 1 or 2 requests the file from server. 3. Server Transfers the file to the requested Client Machine 1 or 2. 4. The source and Destination Addresses are visible in Wireshark along with the actual data which is not encrypted.

Peer-to-Peer Model:

The Peer-to-Peer model is setup using the IPFS. The IPFS protocol is written in GO language. The setup of the experiment is like the Client-Server experiment. 3 virtual machines are used with one machine as the server and the other 2 virtual machines as the client. Figure 11 provides an overview of the model.

The following steps shows how to connect one node to the other node to establish connection between multiple nodes for IPFS Protocol:

1. By using GO language, IPFS can be installed in all the nodes. Once it is installed, the config file in .ipfs folder must be configured as shown below:



```
1 1
2 "API": {
3   "HTTPHeaders": {}
4 },
5 "Addresses": {
6   "API": "/ip4/0.0.0.0/tcp/5001",
7   "Announce": [],
8   "Gateway": "/ip4/0.0.0.0/tcp/8080",
9   "NoAnnounce": [],
10  "Swarm": [
11    "/ip4/0.0.0.0/tcp/4001",
12    "/ip6:::/tcp/4001",
13    "/ip4/0.0.0.0/udp/4001/quic",
14    "/ip6:::/udp/4001/quic"
15  ]
16 },
17 "..."
```

Figure 8-Configuration of config file in .ipfs folder

2. Then the swarm key is generated in Server Machine and that will be copied to other nodes which makes all nodes having the same key. The following figures show the commands for the swarm key are:

go get -u github.com/Kubuxu/go-ipfs-swarm-key-gen/ipfs-swarm-key-gen

ipfs-swarm-key-gen & > ~/.ipfs/swarm.key

A swarm will be generated in the .ipfs folder and that will be copied into the other nodes. In this way, the communication will established without conflicts.

3. The following is used to remove all existing nodes in the ipfs network:
ipfs bootstrap rm -all
4. Every node will be started with the command "ipfs daemon". The Figure 2 in Appendix section represents the function in Server machine.
5. The command "ipfs id" shows all the details about the node. It shows the id of the node which is used to connect with other nodes. The Figures (3) (4) (5) in Appendix

Section shows the details for all the nodes in the ipfs network which will be connected next.

6. Each machine has a unique id which is used to join nodes. The command “ipfs bootstrap add” is used to add the ipfs nodes to one another to form the private IPFS network. The figure (6) in Appendix Section represents the function in Server Machine (eleks, 2021) (Tutorials, 2018).

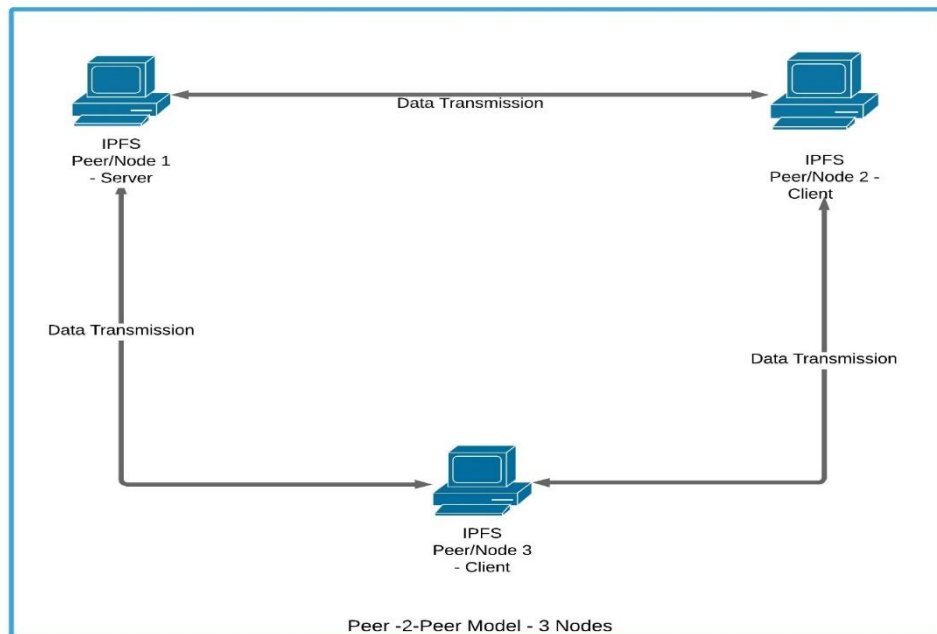


Figure 9-Peer-to-Peer Model using IPFS Protocol

Roadmap:

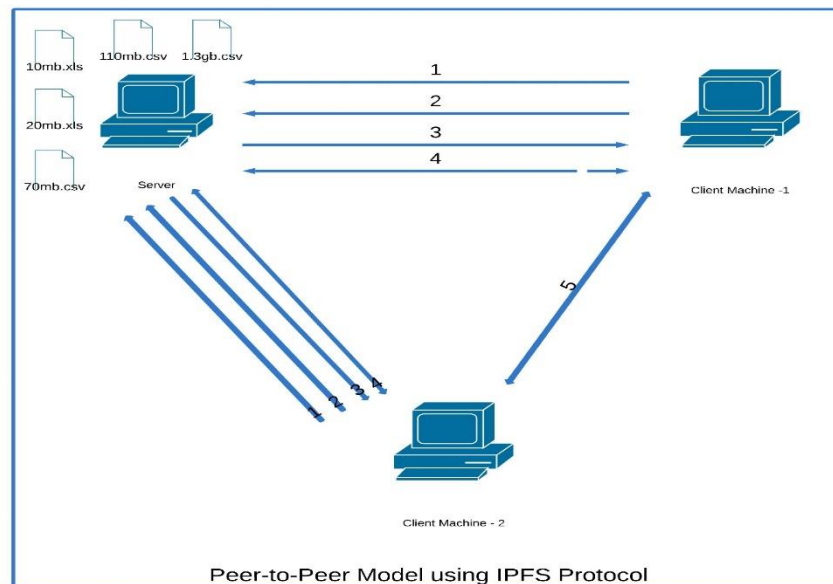
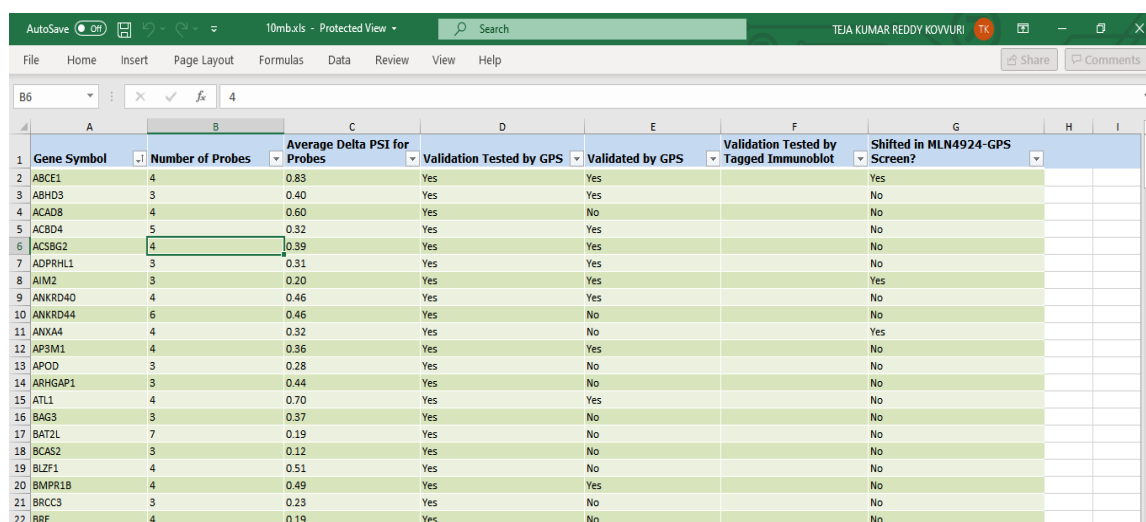


Figure 10-Peer-to-Peer Model using IPFS Protocol Roadmap - 1) Client Machine connects with Server Machine. 2) Client Machine requests the file from server. 3) Server Sends the Requested file (encrypted). The Server Broadcasts the data to all Nodes in the network. 4) Source and Destination addresses are not identifiable in Wireshark as all nodes participate in data transfer. 5) Clients Communicate with each other.

Artefact Development:

The purpose of the experiment is to evaluate the weakness in the existing client-server model and overcoming that with the help of a Peer – to – Peer model protocol named IPFS. By using this protocol, the nodes in a Peer – to – Peer network does not provide the actual data and the destination to where the data is getting transferred. This helps in maintaining the anonymity during the and prevents the recon phase of the Advanced Persistent Threats.

Experiments are conducted on the Client-Server model and the Peer-to-Peer model. The files which were chosen for this experiment are 5 excel files for easy recognition and understanding of data during analysis in Wireshark. The data in the files consist of columns and rows which represent data belonging to a particular entity in each row. 10mb.xls. Excel files maintain a lot of data which is easy to track and find during the data transfer. This is because each data is organized in the form of columns and rows and searching to find a particular word is very easy. As data is transferred in the form of small bits which are equivalent to the words, it will be easy to recognise them and find the same in the excel files.



	A	B	C	D	E	F	G
	Gene Symbol	Number of Probes	Average Delta PSI for Probes	Validation Tested by GPS	Validated by GPS	Validation Tested by Tagged Immunoblot	Shifted in MLN4924-GPS Screen?
1	ABCE1	4	0.83	Yes	Yes		Yes
2	ABHD3	3	0.40	Yes	Yes		No
3	ACAD8	4	0.60	Yes	No		No
4	ACBD4	5	0.32	Yes	Yes		No
5	ACSBG2	4	0.39	Yes	Yes		No
6	ADPRHL1	3	0.31	Yes	Yes		No
7	AIM2	3	0.20	Yes	Yes		Yes
8	ANKRD40	4	0.46	Yes	Yes		No
9	ANKRD44	6	0.46	Yes	No		No
10	ANXA4	4	0.32	Yes	No		Yes
11	AP3M1	4	0.36	Yes	Yes		No
12	APOD	3	0.28	Yes	No		No
13	ARHGAP1	3	0.44	Yes	No		No
14	ATL1	4	0.70	Yes	Yes		No
15	BAG3	3	0.37	Yes	No		No
16	BAT2L	7	0.19	Yes	No		No
17	BCAS2	3	0.12	Yes	No		No
18	BLZF1	4	0.51	Yes	No		No
19	BMPRI1B	4	0.49	Yes	Yes		No
20	BRCC3	3	0.23	Yes	No		No
21	BRE	4	0.19	Yes	No		No

Figure 11-10mb.xls

The Appendix Section has the remaining figures for 20mb.xls, 70mb.csv, 110mb.csv, and 1.3gb.csv files.

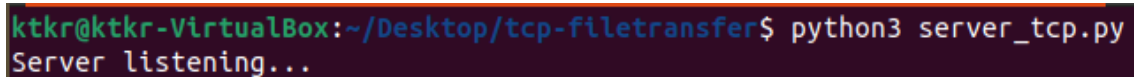
Experiment on Client - Server model

The server starts and listens for connections from clients. Once the clients get connected, the file transfer between the nodes take place as per the request of the client. The server accepts connection from each client at a time and finishes the transmission of data before accepting connection from another client.

Wireshark is used to consolidate the traffic and the results will be analysed. Further experiments are evaluated. The Figure-5 representing Wireshark in Kali Linux can be seen in Appendix section.

The Procedure for the experiment

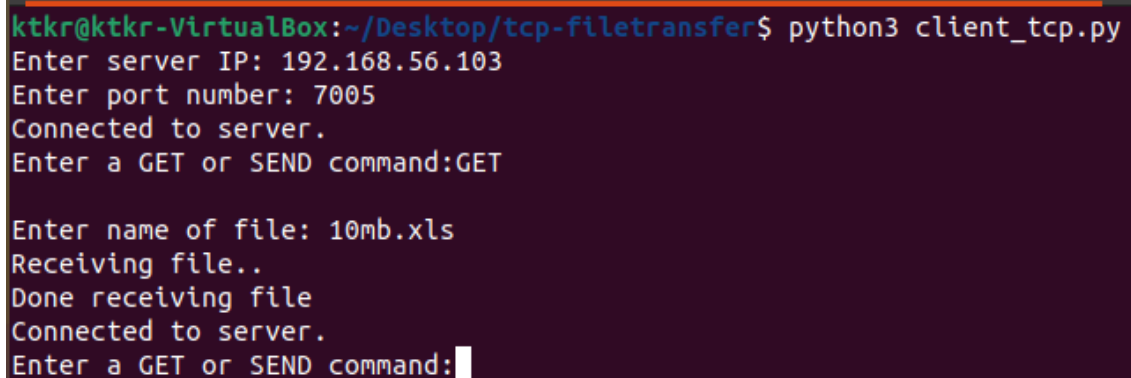
1. Server is configured to start and listen for client to establish a connection. When the connection with the client is established, the server sends the requested file to the client.



```
ktkr@ktkr-VirtualBox:~/Desktop/tcp-filetransfer$ python3 server_tcp.py
Server listening...
```

Figure 12-Python Client Server model - Server machine

2. Client is started and waiting to establish connection with server. In the Client Machine, the server IP address and the port number are supplied. In addition to that, the method to request the file which is the GET method will also be supplied. Finally, the client gives the file name.



```
ktkr@ktkr-VirtualBox:~/Desktop/tcp-filetransfer$ python3 client_tcp.py
Enter server IP: 192.168.56.103
Enter port number: 7005
Connected to server.
Enter a GET or SEND command:GET

Enter name of file: 10mb.xls
Receiving file..
Done receiving file
Connected to server.
Enter a GET or SEND command:
```

Figure 13-Python Client Server model - Client Machine-1

The server searches for the requested file. Once the file is found, the file will be sent by the server. The following figure shows that the file has been successfully sent by the server to the client. The client receives the file. If in case the server does not find the file, it displays no such file or directory available message.

```

ktr@ktr-VirtualBox:~/Desktop/tcp-filetransfer$ python3 server_tcp.py
Server listening...
127.0.1.1
Got connection from ('192.168.56.104', 52926)
Awaiting command from client...
Received GET command from client. Waiting for filename.
Sending file...
Done sending
Got connection from ('192.168.56.104', 52928)
Awaiting command from client...

```

Figure 14-Server Machine sending data

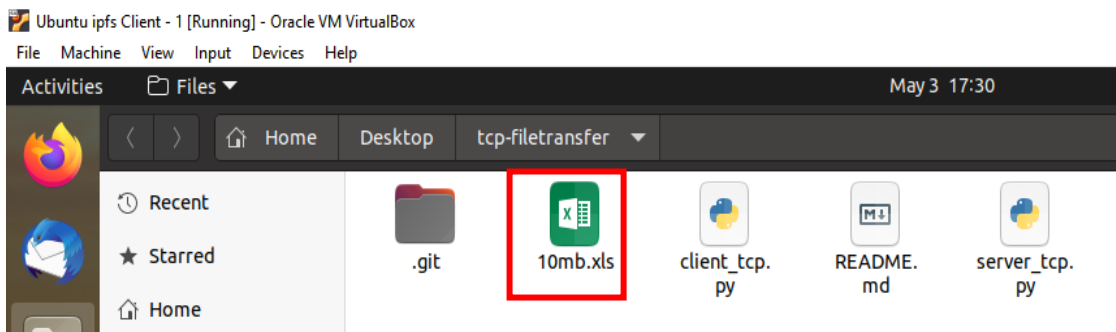


Figure 15-Transferred file visible in Client Machine-1

Result

The excel files used in the experiment are transferred from the server to the client. There is a total of 5 excel files used. A certain key word is selected from the 5 excel files as identity for the excel file. Figure 8 shows the key word “WDR67” is used. The key words for the other excel files are revealed in figures (8) (10) (12) (14) in the appendix section.

263	WDR4	7	0.26	yes	yes	yes	no
264	WDR41	3	0.21	Yes	Yes		No
265	WDR49	1	0.51	Yes	Yes		No
266	WDR67	4	0.55	Yes	No		No
267	WDR68	4	0.21	Yes	Yes		No

Figure 16-“WDR67” in 10mb.xls is selected

The traffic between server and client has been captured in Wireshark. The payload in the traffic reveals the actual keyword selected in the selected excel files. In client-server model, the concept relies on encryption tool such as SSL or TLS to hide the content of the traffic. These results are shown in figure 9. The result for other excel files are available in Figure (7) (9) (11) (13) in appendix.



Figure 17-Actual data visibility in Wireshark

The table below provides additional information of the experiment for the client-server model.

File	Client Name	Destination address	Data visibility	Sample data	Data identification in traffic
10mb.xls	Ubuntu ipfs Client – 1	192.168.56.104	Yes	WDR67	Yes
20mb.xls	Ubuntu ipfs Client – 2	192.168.56.105	Yes	149.29	Yes
1.3gb.csv	Ubuntu ipfs Client – 1	192.168.56.104	Yes	evie.hamby@gmail.com	Yes
110mb.csv	Ubuntu ipfs Client – 2	192.168.56.105	Yes	44005	Yes
70mb.csv	Ubuntu ipfs Client – 1	192.168.56.104	Yes	Global Rank, Tld Rank	Yes

Table 6-Result and information for the Client-Server model experiment.

This experiment shows that there is no security, and anyone can see the plain text using tools like Wireshark. An adversary can know to which node/address the data has been passed. This leads to more several attacks by adversary on the communication.

Experiment on Peer – to - Peer model

The same experiments have been conducted on the same nodes using IPFS protocol. In this experiment, the traffic will be analyzed to see any actual data or to identify the destination to where the data is transferred.

The Procedure for the experiment

In the server, IPFS stores the files in its database and generates a unique hash value for the file. The “add” command is used to add file into the database. It is used to upload the 5 excel files into the database. Figure 12 shows how the command is used for the uploading of the excel file.

```
ktkr@ktkr-VirtualBox:~/Desktop/tcp-filetransfer$ ipfs add 10mb.xls
added QmZzF5r3KPmyjFM7gPeCoHeDWscGeHS1x7eJLSM66SfQRA 10mb.xls
10.18 MiB / 10.18 MiB [=====] 100.00%
ktkr@ktkr-VirtualBox:~/Desktop/tcp-filetransfer$
```

Figure 18-Peer-to-Peer Model Server Machine using IPFS Protocol

In Client, the “get” command is used to send a request to the ipfs server/node for the file. The example of the command used to request for the file can be seen in figure 14.

```
ktkr@ktkr-VirtualBox:~/Desktop/tcp-filetransfer/ipfs$ ipfs get QmZzF5r3KPmyjFM7gPeCoHeDWscGeHS1x7eJLSM66SfQRA
Saving file(s) to QmZzF5r3KPmyjFM7gPeCoHeDWscGeHS1x7eJLSM66SfQRA
10.18 MiB / 10.18 MiB [=====] 100.00% 1s
ktkr@ktkr-VirtualBox:~/Desktop/tcp-filetransfer/ipfs$
```

Figure 19-Peer-to-Peer Model Client Machine-1 using IPFS Protocol

Result

With the help of Wireshark, the entire data transmission has been captured and analyzed.

192.168.56.103	192.168.56.104	TCP	108	4001 → 4001	[PSH, ACK] Seq=
192.168.56.104	192.168.56.103	TCP	254	4001 → 4001	[PSH, ACK] Seq=
192.168.56.105	192.168.56.104	TCP	107	4001 → 4001	[PSH, ACK] Seq=
192.168.56.104	192.168.56.103	TCP	87	4001 → 4001	[PSH, ACK] Seq=
192.168.56.103	192.168.56.104	TCP	66	4001 → 4001	[ACK] Seq=10723
192.168.56.104	192.168.56.105	TCP	254	4001 → 4001	[PSH, ACK] Seq=
192.168.56.103	192.168.56.104	TCP	87	4001 → 4001	[PSH, ACK] Seq=
192.168.56.105	192.168.56.104	TCP	108	4001 → 4001	[PSH, ACK] Seq=

Figure 20-Analysis of Traffic using Wireshark in Kali Linux

In IPFS, the sender uses the broadcasting method to communicate with other node. The actual data is transferred to all nodes and the legitimate node will receive the data. Other node rejects the data. This helps in maintaining the anonymity. It makes tracking of traffic difficult as this is demonstrated in figure 15. The source and destination are unknown.

IPFS encrypts the data before it is being transmitted to the other nodes. The diagram in figure 16 shows the content is not readable. This helps to prevent the content in the payload of the traffic to be exposed in the process of man in the middle attack. Hence, this enhances the security of transmission and keep the network safe from adversary.

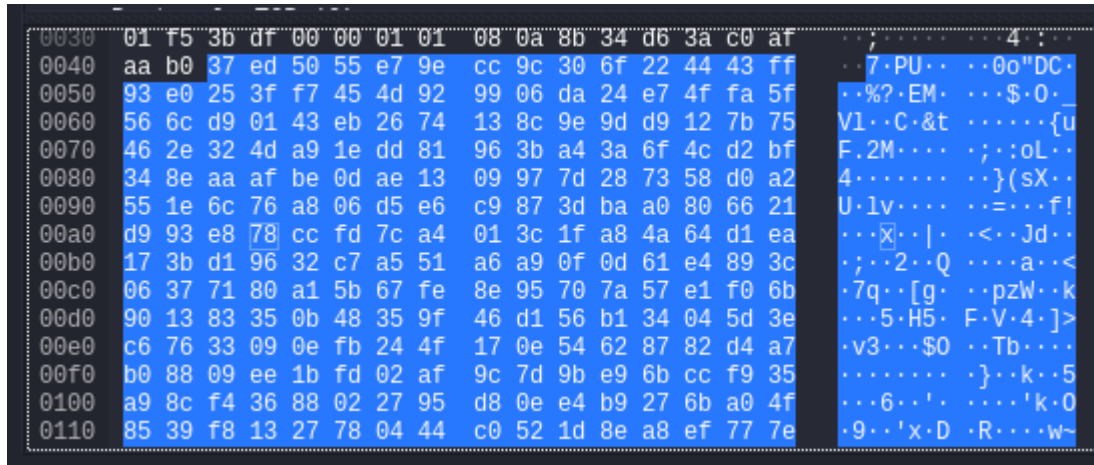


Figure 21-Data Analysis in Wireshark - Data is Encrypted

The experiment is repeated for the 5 excel files and the results are noted in table 2. In table 2, the size of the file does not affect the result of the network.

Sample	Client	Destination address	Data visibility	Sample data	Original data identification in traffic
10mb.xls	Ubuntu ipfs Client – 1	Unknown	NO	WDR67	NO
20mb.xls	Ubuntu ipfs Client – 2	Unknown	NO	149.29	NO
1.3gb.csv	Ubuntu ipfs Client – 2	Unknown	NO	evie.hamby@gmail.com	NO
110mb.csv	Ubuntu ipfs Client – 2	Unknown	NO	44005	NO
70mb.csv	Ubuntu ipfs Client – 1	Unknown	NO	Global Rank, Tld Rank	NO

Table 7-Result and information for the Peer-to-Peer model experiment.

Evaluation

Summary Table for comparison of Client-Server model and Peer-to-Peer model:

Type of Model	Sample	Client	Destination Address	Data visibility	Sample Data	Original data identification in traffic
Client Server Model	10mb.xls	Ubuntu ipfs Client – 1	192.168.56.104	Yes	WDR67	Yes
	20mb.xls	Ubuntu ipfs Client – 2	192.168.56.105	Yes	149.29	Yes
	1.3gb.csv	Ubuntu ipfs Client – 2	192.168.56.104	Yes	evie.hamby@gmail.com	Yes
	110mb.csv	Ubuntu ipfs Client – 2	192.168.56.105	Yes	44005	Yes
	70mb.csv	Ubuntu ipfs Client – 1	192.168.56.104	Yes	Global Rank, Tld Rank	Yes
Peer-to-Peer Model	10mb.xls	Ubuntu ipfs Client – 1	Unknown	NO	WDR67	NO

	20mb.xls	Ubuntu ipfs Client – 2	Unknown	NO	149.29	NO
	1.3gb.csv	Ubuntu ipfs Client – 2	Unknown	NO	evie.hamby@gmail.com	NO
	110mb.csv	Ubuntu ipfs Client – 2	Unknown	NO	44005	NO
	70mb.csv	Ubuntu ipfs Client – 1	Unknown	NO	Global Rank, Tld Rank	NO

Table 8-Comparison of the results of Client-Server Model and Peer-to-Peer Model

When the server machine using IPFS Protocol in Peer-to-Peer Model sends a file to all nodes, all the nodes receive the same data in encrypted form. The data also varies from node to node. The server sends data which is encrypted differently for each node.

Future Work

With the help of the above experiments, it is evident that the source and destination addresses are autonomous with the use of IPFS protocol which protects the nodes from adversaries. Further experiments can be conducted to find more features which are helpful for Peer-to-Peer model and strengthens the security for the nodes.

References

- ALSHAMRANI, A., MYNENI, S., CHOWDHARY, A. & HUANG, D. 2019. A Survey on Advanced Persistent Threats: Techniques, Solutions, Challenges, and Research Opportunities. *IEEE Communications Surveys & Tutorials*, 21, 1851-1877.
- BRAHIM ID MESSAOUD, K. G., MOHAMED WAHBI, MOHAMED SADIK 2016. <Advanced Persistent Threat - new analysis.pdf>.
- CYBERAGENTS. 2017. *Peer-to-Peer Networks* [Online]. Available: <https://www.cyberagentsinc.com/2018/09/14/peer-to-peer-networks/> [Accessed 18 May 2021].
- DEV BHATNAGAR, S. S., SUNIL KUMAR KHATRI 2019. <Advance Persistent Threat and Cyber Spying - The Big Picture,.pdf>.
- DUCHESSI, P. & CHENGALUR-SMITH, I. 1998. Client/server benefits, problems, best practices. *Communications of the ACM*, 41, 87-94.
- ELEKS. 2021. *IPFS Tutorial: Building a Private IPFS Network with IPFS-Cluster for Data Replication* [Online]. Available: <https://labs.eleks.com/2019/03/ipfs-network-data-replication.html> [Accessed 21 May 2021].
- ETHEREUM, G. 2019. *Overview | Go Ethereum* [Online]. Available: <https://geth.ethereum.org/docs/whisper/whisper-overview#:~:text=Whisper%20is%20a%20pure%20identity,the%20underlying%20C3%90%CE%9EVp2p%20Wire%20Protocol>. [Accessed 30 May 2021].
- HUANG, H.-S., CHANG, T.-S. & WU, J.-Y. 2020. A Secure File Sharing System Based on IPFS and Blockchain. *Proceedings of the 2020 2nd International Electronics Communication Conference*.
- INFURA. 2021. *An Introduction to IPFS (Interplanetary File System) | Infura Blog | Tutorials, Case Studies, News, Feature Announcements* [Online]. Available: <https://blog.infura.io/an-introduction-to-ipfs/#:~:text=At%20its%20core%2C%20IPFS%20is,%2C%20TOR%2C%20and%20even%20Bluetooth>. [Accessed 22 May 2021].
- IPFS. 2021a. *Configure a node | IPFS Docs* [Online]. Available: <https://docs.ipfs.io/how-to/configure-node/#discovery> [Accessed 22 May 2021].
- IPFS. 2021b. *How IPFS works | IPFS Docs* [Online]. Available: <https://docs.ipfs.io/concepts/how-ipfs-works/#content-addressing> [Accessed 30 May 2021].
- JAVOID, N. 2019. *Data sharing on IPFS by owner* [Online]. Available: https://www.researchgate.net/figure/Data-sharing-on-IPFS-by-owner_fig1_335652136 [Accessed 18 May 2021].
- JOANNA MOUBARAKA, M. C., ERIC FILIOL 2020. *On distributed ledgers security and illegal uses*.
- LOO, A. W. 2003. The future of peer-to-peer computing. *Communications of the ACM*, 46, 56-61.

- MARTIN USSATH, D. J., FENG CHENG, CHRISTOPH MEINEL 2016. <Advanced Persistent Threats -Behind the Scenes.pdf>.
- OLUWATOSIN, H. S. 2014. <Client-Server Model.pdf>.
- ROOMI, M. 2020, November 15. *7 Advantages and Disadvantages of Peer to Peer Network | Drawbacks & Benefits of Peer to Peer Network* [Online]. Available: <https://www.hitechwhizz.com/2020/11/7-advantages-and-disadvantages-drawbacks-benefits-of-p2p-network.html> [Accessed 29 March 2021].
- RTI. 2015. *Network Communications Models* [Online]. Available: https://community.rti.com/static/documentation/connex-dds/5.2.0/doc/manuals/connex-dds/html/files/RTI_ConnextDDS_CoreLibraries_GettingStarted/Content/UsersManual/Network_Communications_Models.htm [Accessed 18 May 2021].
- S VENU GOPAL, N. S. R., S K LOKESH NAIK 2016. <Dynamic Sharing of Files from Disconnected Nodes.pdf>.
- SAWSAN ALI HAMID, R. A. A., DR.RUAA ALI KHAMEES 2020. <What is Client-Server System_ Architecture, Issues and Challenge.pdf>.
- STEFAN KRAXBERGER, U. P. 2009. <Security Concept for Peer-to-Peer Systems.pdf>.
- SUN, T. N., TEODOROV, C. & ROUX, L. L. 2020. Operational design for advanced persistent threats. *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*.
- TEAM, I. 2019. *Advanced Persistent Threat: Challenge Accepted* [Online]. Available: <https://blog.teamascend.com/advanced-persistent-threat> [Accessed 18 May 2021].
- TECHNOLOGIES, E. 2021. *Decentralized Application Messaging with Whisper — Part 1* [Online]. Available: <https://blog.enumai.io/update/2018/08/08/decentralized-application-messaging-with-whisper.html> [Accessed 30 May 2021].
- TOMU, D. 2018. *Whisper — Shh!. What if we could send an encrypted... | by David Tomu | Caelum Labs | Medium* [Online]. Available: <https://medium.com/caelumlabs/whisper-shh-bc5416ec0046> [Accessed 30 May 2021].
- IPFS Tutorials #4 - How to setup P2P Private Network IPFS*, 2018. Directed by TUTORIALS, D.
- ZHANG, L., ZHANG, Z., JIN, Z., SU, Y. & WANG, Z. 2020. An approach of covert communication based on the Ethereum whisper protocol in blockchain. *International Journal of Intelligent Systems*, 36, 962-996.
- ZHANG, L., ZHANG, Z., JIN, Z., SU, Y. & WANG, Z. 2021. An approach of covert communication based on the Ethereum whisper protocol in blockchain. *International Journal of Intelligent Systems*, 36, 962-996.

Appendix

192.168.56.104	239.255.255.250	SSDP	136 M-SEARCH * HTTP/1.1
192.168.56.104	224.0.0.22	IGMPv3	60 Membership Report / Join group 239.255.255.250 for any sources
192.168.56.104	224.0.0.22	IGMPv3	60 Membership Report / Join group 239.255.255.250 for any sources
fe80::a2be:79f4:2a1...	ff02::fb	MDNS	106 Standard query 0xb0a5 PTR _ipfs-discovery._udp.local, "QM" question
192.168.56.104	224.0.0.22	IGMPv3	60 Membership Report / Leave group 239.255.255.250
192.168.56.103	224.0.0.22	IGMPv3	60 Membership Report / Join group 239.255.255.250 for any sources
192.168.56.103	239.255.255.250	SSDP	136 M-SEARCH * HTTP/1.1
fe80::fca5:16f8:662...	ff02::fb	MDNS	106 Standard query 0x77b4 PTR _ipfs-discovery._udp.local, "QM" question
fe80::a2be:79f4:2a1...	ff02::1:ff23:2686	ICMPv6	86 Neighbor Solicitation for fe80::fca5:16f8:6623:2686 from 08:00:27:eb:ec:2a
fe80::fca5:16f8:662...	fe80::a2be:79f4:2a1...	ICMPv6	86 Neighbor Advertisement fe80::fca5:16f8:6623:2686 (sol, ovr) is at 08:00:27:88:58:61
fe80::a2be:79f4:2a1...	fe80::fca5:16f8:662...	MDNS	327 Standard query response 0x0000 PTR 12D3KooWBCWrzgsSF7zmuo7a3ANZ5aPsvFhwbtGhrESVSeVuGhFn._ipfs-disc

Figure 1-MDNS in IPFS Protocol connection establishment

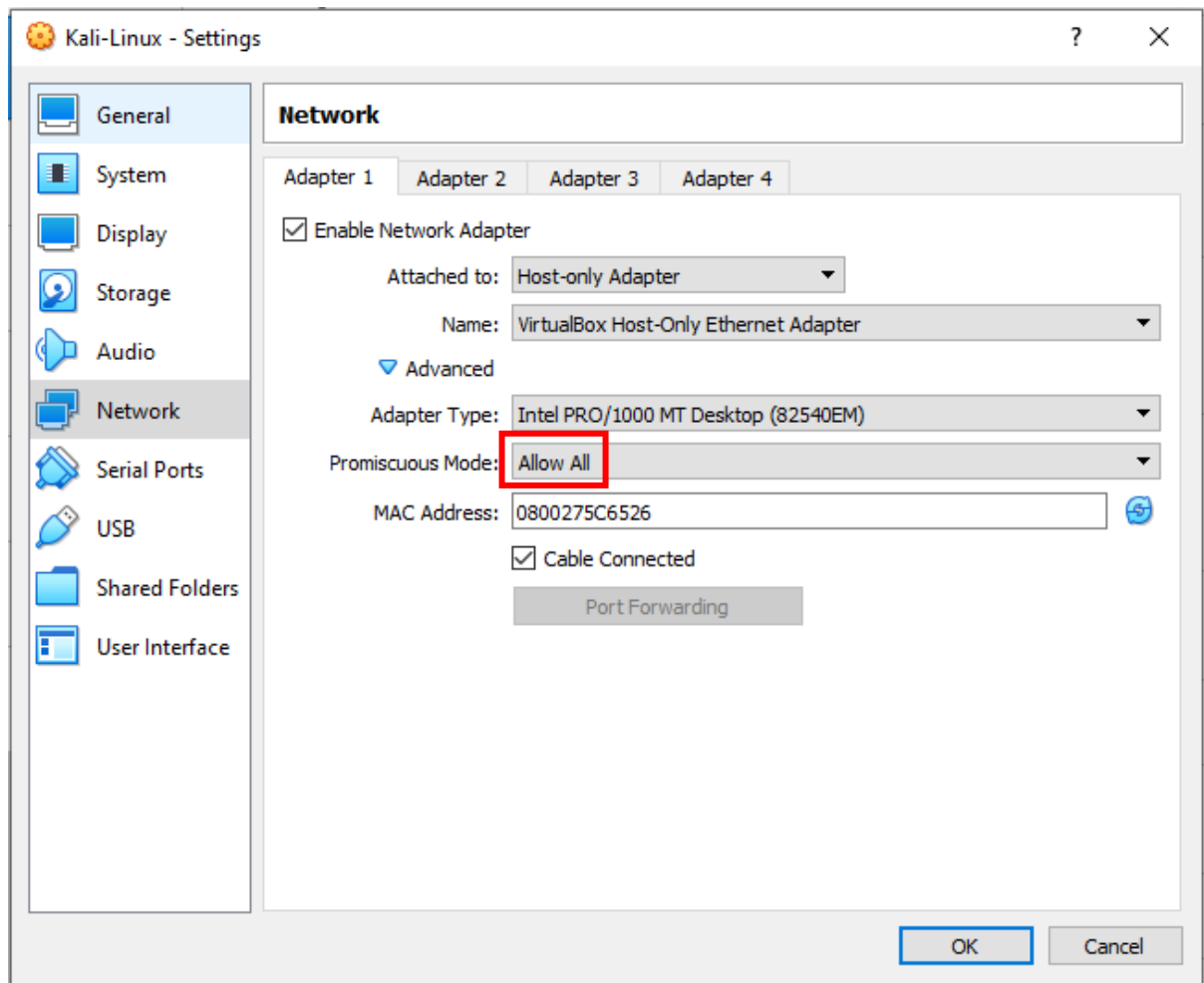
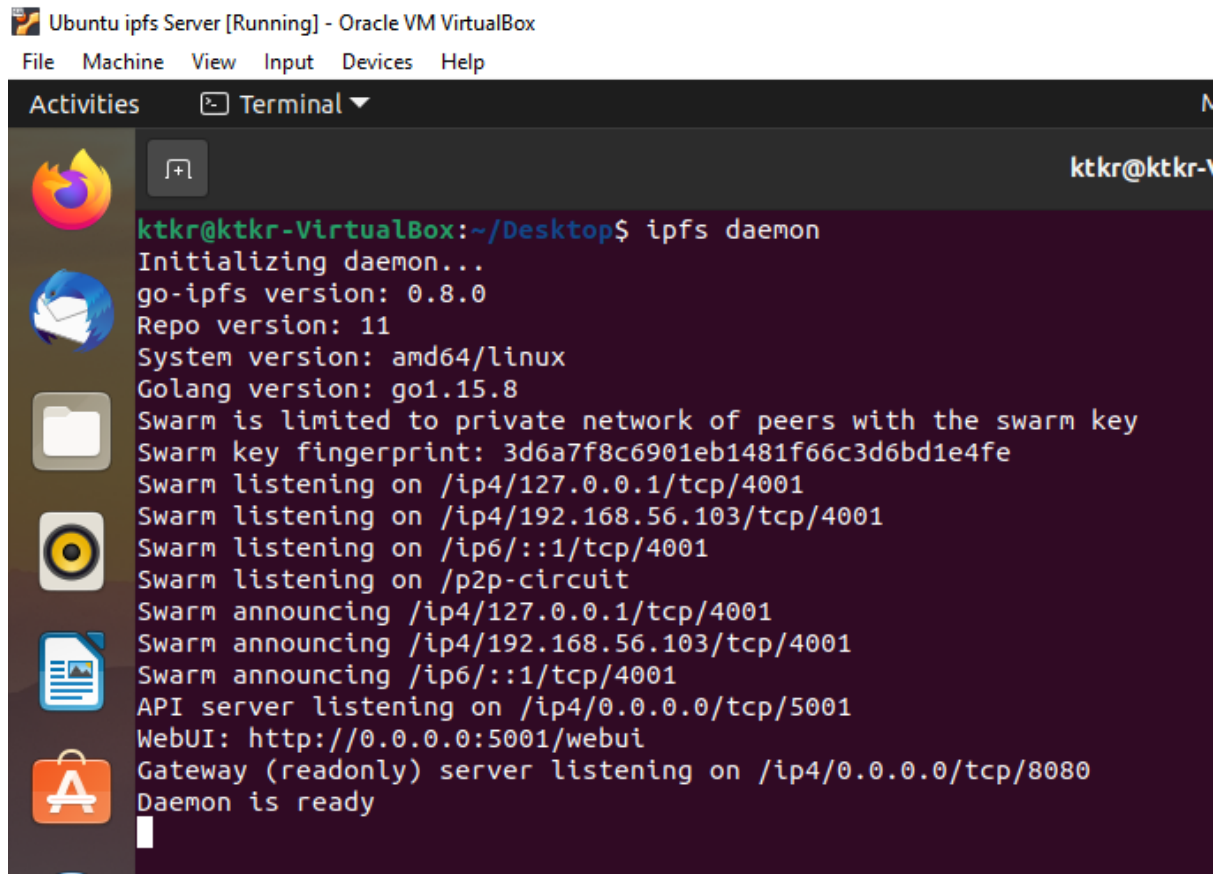
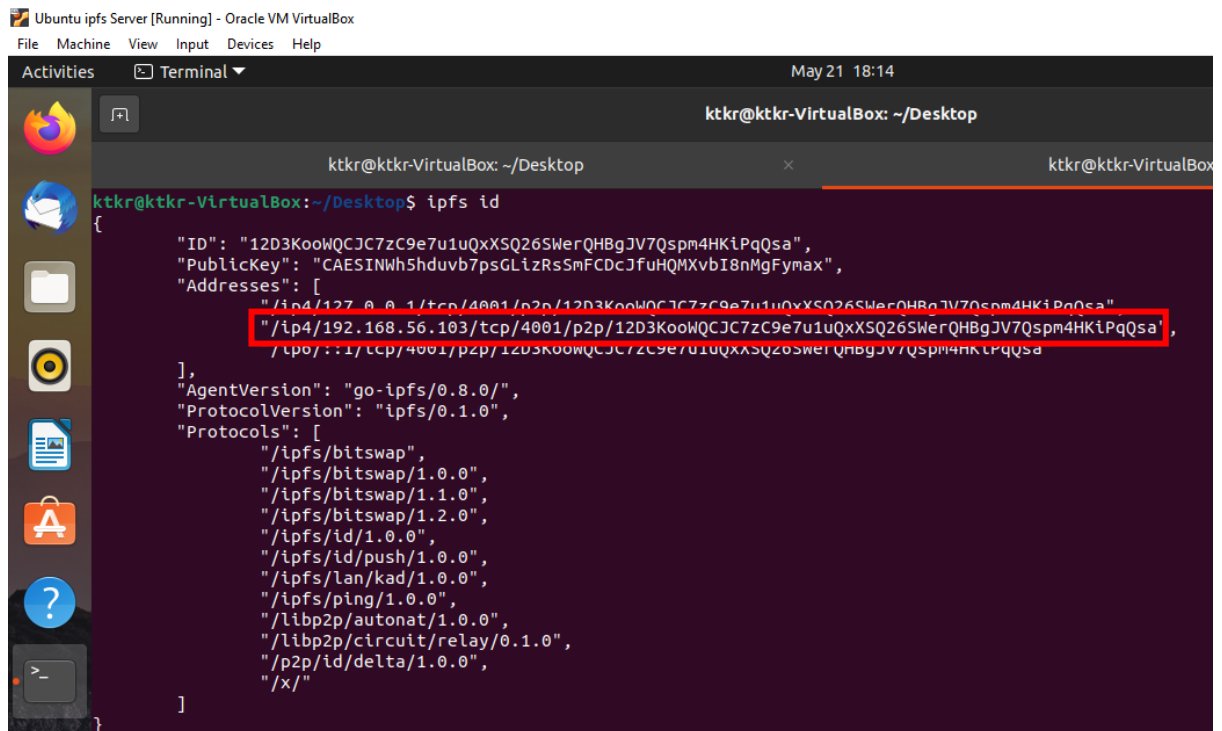


Figure 2-Promiscuous Mode-Allow All in Network Settings of Virtual Box for Kali Linux Virtual Machine



```
ktkr@ktkr-VirtualBox:~/Desktop$ ipfs daemon
Initializing daemon...
go-ipfs version: 0.8.0
Repo version: 11
System version: amd64/linux
Golang version: go1.15.8
Swarm is limited to private network of peers with the swarm key
Swarm key fingerprint: 3d6a7f8c6901eb1481f66c3d6bd1e4fe
Swarm listening on /ip4/127.0.0.1/tcp/4001
Swarm listening on /ip4/192.168.56.103/tcp/4001
Swarm listening on /ip6:::1/tcp/4001
Swarm listening on /p2p-circuit
Swarm announcing /ip4/127.0.0.1/tcp/4001
Swarm announcing /ip4/192.168.56.103/tcp/4001
Swarm announcing /ip6:::1/tcp/4001
API server listening on /ip4/0.0.0.0/tcp/5001
WebUI: http://0.0.0.0:5001/webui
Gateway (readonly) server listening on /ip4/0.0.0.0/tcp/8080
Daemon is ready
```

Figure 3-ipfs daemon in Server Machine



```
ktkr@ktkr-VirtualBox:~/Desktop$ ipfs id
{
  "ID": "12D3KooWQCJC7zC9e7u1uQxXSQ26SWerQHBgJV7Qspm4HKiPqQsa",
  "PublicKey": "CAESINWhShduvb7psGLtzRsSmFCDCJFuHQMxvbI8nMgFymax",
  "Addresses": [
    "/ip4/127.0.0.1/tcp/4001/p2p/12D3KooWQCJC7zC9e7u1uQxXSQ26SWerQHBgJV7Qspm4HKiPqQsa",
    "/ip4/192.168.56.103/tcp/4001/p2p/12D3KooWQCJC7zC9e7u1uQxXSQ26SWerQHBgJV7Qspm4HKiPqQsa",
    "/ip6:::1/tcp/4001/p2p/12D3KooWQCJC7zC9e7u1uQxXSQ26SWerQHBgJV7Qspm4HKiPqQsa"
  ],
  "AgentVersion": "go-ipfs/0.8.0/",
  "ProtocolVersion": "ipfs/0.1.0",
  "Protocols": [
    "/ipfs/bitswap",
    "/ipfs/bitswap/1.0.0",
    "/ipfs/bitswap/1.1.0",
    "/ipfs/bitswap/1.2.0",
    "/ipfs/id/1.0.0",
    "/ipfs/id/push/1.0.0",
    "/ipfs/lan/kad/1.0.0",
    "/ipfs/ping/1.0.0",
    "/libp2p/autonat/1.0.0",
    "/libp2p/circuit/relay/0.1.0",
    "/p2p/id/delta/1.0.0",
    "/x/"
  ]
}
```

Figure 4-ipfs details for Server Machine


```
Ubuntu ipfs Client - 1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal May 21 18:16
ktkr@ktkr-VirtualBox: ~/Desktop

ktkr@ktkr-VirtualBox: ~/Desktop$ ipfs id
{
  "ID": "12D3KooWBCWrzgsSF7zmuo7a3ANZ5aPsvFhwbtGHRiESVSeVuGhFn",
  "PublicKey": "CAESIbSIS4fzqIf3JrYg6lFXLUfcDC6MiTFHLXKwumNEyo3R",
  "Addresses": [
    "/ip4/127.0.0.1/tcp/4001/p2p/12D3KooWBCWrzgsSF7zmuo7a3ANZ5aPsvFhwbtGHRiESVSeVuGhFn",
    "/ip4/192.168.56.104/tcp/4001/p2p/12D3KooWBCWrzgsSF7zmuo7a3ANZ5aPsvFhwbtGHRiESVSeVuGhFn",
    "/ip6::1/tcp/4001/p2p/12D3KooWBCWrzgsSF7zmuo7a3ANZ5aPsvFhwbtGHRiESVSeVuGhFn"
  ],
  "AgentVersion": "go-ipfs/0.8.0/",
  "ProtocolVersion": "ipfs/0.1.0",
  "Protocols": [
    "/ipfs/bitswap",
    "/ipfs/bitswap/1.0.0",
    "/ipfs/bitswap/1.1.0",
    "/ipfs/bitswap/1.2.0",
    "/ipfs/id/1.0.0",
    "/ipfs/id/push/1.0.0",
    "/ipfs/lan/kad/1.0.0",
    "/ipfs/ping/1.0.0",
    "/libp2p/autonat/1.0.0",
    "/libp2p/circuit/relay/0.1.0",
    "/p2p/id/delta/1.0.0",
    "/x/"
  ]
}
```

Figure 5-ipfs details for Ubuntu ipfs Client -1

```
Ubuntu ipfs Client - 2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal May 21 18:40
ktkr@ktkr-VirtualBox: ~/Desktop

ktkr@ktkr-VirtualBox: ~/Desktop$ ipfs id
{
  "ID": "12D3KooWR8F4WDzYBaJmVCuikkNgu9FTobqbnBHjJNivqtmNnAc",
  "PublicKey": "CAESIOnZuM3EZu+88XW8elgJNKZ1BqStsl8yZG+exU5IPHvp",
  "Addresses": [
    "/ip4/127.0.0.1/tcp/4001/p2p/12D3KooWR8F4WDzYBaJmVCuikkNgu9FTobqbnBHjJNivqtmNnAc",
    "/ip4/192.168.56.105/tcp/4001/p2p/12D3KooWR8F4WDzYBaJmVCuikkNgu9FTobqbnBHjJNivqtmNnAc",
    "/ip6::1/tcp/4001/p2p/12D3KooWR8F4WDzYBaJmVCuikkNgu9FTobqbnBHjJNivqtmNnAc"
  ],
  "AgentVersion": "go-ipfs/0.8.0/",
  "ProtocolVersion": "ipfs/0.1.0",
  "Protocols": [
    "/ipfs/bitswap",
    "/ipfs/bitswap/1.0.0",
    "/ipfs/bitswap/1.1.0",
    "/ipfs/bitswap/1.2.0",
    "/ipfs/id/1.0.0",
    "/ipfs/id/push/1.0.0",
    "/ipfs/lan/kad/1.0.0",
    "/ipfs/ping/1.0.0",
    "/libp2p/autonat/1.0.0",
    "/libp2p/circuit/relay/0.1.0",
    "/p2p/id/delta/1.0.0",
    "/x/"
  ]
}
```

Figure 6-ipfs details of Ubuntu ipfs Client – 2

AutoSave Off

File Home Insert Page Layout

Paste

Clipboard

Font

Calibri 11 A

B I U

A1

Region

	A	B	C	D
1	Region	Product	Date	Sales
2	West	Prod T	#####	53395.18
3	West	Prod K	#####	116609.7
4	South	Prod F	#####	72524.1
5	South	Prod J	#####	22538.48
6	North	Prod D	#####	45616.53
7	North	Prod J	#####	81902.36
8	South	Prod T	#####	72000.37
9	North	Prod O	#####	99227.68
10	West	Prod A	#####	93520.55
11	South	Prod J	#####	77445.83
12	South	Prod E	#####	38690.91
13	South	Prod G	#####	81105.31
14	East	Prod A	#####	66631.66
15	South	Prod F	#####	26633.68
16	East	Prod O	#####	11441.32
17	West	Prod Q	#####	32539.48
18	East	Prod H	#####	58734.46
19	North	Prod N	#####	56381.12
20	North	Prod R	#####	26348.75
21	North	Prod T	#####	79762.8
22	North	Prod F	#####	94404.9
23	North	Prod I	#####	38908.2

110mb

Figure 7-20mb.xls

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
	GlobalRank	TldRank	Domain	TLD	RefSubNe	RefIPs	IDN_Dom	IDN_TLD	PrevGlob	PrevTldRa	PrevRefSu	PrevRefIPs									
1	1	1	google.co.com		501746	2726950	google.co.com		1	1	501560	2728650									
2	2	2	facebook.com		501216	2906749	facebook.com		2	2	500947	2907145									
3	3	3	youtube.com		457447	2404541	youtube.com		3	3	457243	2405388									
4	4	4	twitter.com		444741	2321701	twitter.com		4	4	444560	2322395									
5	5	5	instagram.com		365880	1790153	instagram.com		5	5	365623	1790231									
6	6	6	linkedin.com		350094	1537426	linkedin.com		6	6	349865	1537650									
7	7	7	microsoft.com		307637	1019838	microsoft.com		7	7	307369	1021930									
8	8	8	apple.com		299300	1035736	apple.com		8	8	298853	1035516									
9	9	9	wikipedia.org		290527	1118887	wikipedia.org		9	1	290419	1119287									
10	10	9	googletag.com		252394	892498	googletag.com		10	9	252288	892593									
11	11	1	youtu.be	be	249645	887014	youtu.be		11	1	249299	886738									
12	12	10	plus.google.com		243570	1002012	plus.google.com		13	10	243370	1002056									
13	13	2	en.wikipedia.org		243470	835676	en.wikipedia.org		12	2	243405	836044									
14	14	11	pinterest.com		233728	984824	pinterest.com		14	11	233702	985712									
15	15	12	vimeo.com		229468	879389	vimeo.com		15	12	229241	879727									
16	16	13	play.google.com		228940	693328	play.google.com		16	13	228777	693210									
17	17	14	maps.google.com		217176	787275	maps.google.com		17	14	217066	787809									
18	18	15	adobe.com		214838	634177	adobe.com		18	15	214547	634155									
19	19	1	google.gl	gl	212396	724409	google.gl		19	1	212376	724623									
20	20	16	itunes.apple.com		206385	579414	itunes.apple.com		21	17	206168	579673									
21	21	17	wordpress.com		206382	710325	wordpress.com		20	16	206236	710383									
22	22				202850	655774	bit.ly		22	1	202802	655772									

Figure 8-70mb.csv

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	Cell Line	pr_gene_id	pr_gene_sym	pr_gene_title	A549	A549	A549	A549	A549	A549	A549	A549	A549	A549	A549
1	Cell Line	pr_gene_id	pr_gene_sym	pr_gene_title	A549	A549	A549	A549	A549	A549	A549	A549	A549	A549	A549
2	Clone ID	na	na	na	BRDN000055	TRCN000046	TRCN000047	TRCN000047	TRCN000048	BRDN000055	BRDN000056	TRCN000046	BRDN000055	BRDN000056	BRDN000056
3	Allele	na	na	na	ABCB9_p.R2	ABCB9_WT	ACAA1_WT	ACAT2_WT	YACD_WT_V5	AKR1B1_p.F2	AKR1B1_WT	AKR1B1_WT	AKT1_p.D323	AKT1_WT	ARAF_p.D42
4	201000_at	16	AARS	alanyl-tRNA s	0.6095	-0.5547	0.5411	-1.34	-0.06833	0.295	0.1039	0.8693	-0.393	1.04	-2.09
5	203192_at	10058	ABCB6	ATP-binding	0.5096	0.1295	0.2769	-1.62	0.4843	0.5115	-0.1192	-0.9948	2.17	3.03	0.2669
6	209380_s_at	10057	ABCC5	ATP-binding	-0.8067	0.2974	-0.515	0.3597	-0.4837	-0.453	-1.2	-0.4544	0.08655	2.2	-0.8352
7	200045_at	23	ABCF1	ATP-binding	-0.9434	2.33	-0.2477	2.12	-0.8212	-0.1686	0.588	-0.7175	1.06	0.3724	0.002513
8	202394_s_at	55324	ABCF3	ATP-binding	-0.5631	-0.5855	-0.3076	-1.46	-0.1111	0.349	-0.3765	0.1804	2.13	0.4604	-1.55
9	218581_at	63874	ABHD4	abhydrolase	-0.3787	-0.6788	0.2574	1.37	-0.9085	1.46	1.72	1.46	1.23	-0.5994	2.15
10	221552_at	57406	ABHD6	abhydrolase	0.2857	0.6382	-0.6588	-0.3746	-1.09	-1.01	-0.2272	-1.16	0.8078	-0.1902	-2.69
11	202123_s_at	25	ABL1	c-abl oncogene	-0.4497	0.8127	0.1201	0.286	-0.1833	0.6405	0.8823	-0.6508	0.115	1.35	1.32
12	214274_s_at	30	ACAA1	acetyl-CoA a	-0.3984	-1.35	20.87	-0.8874	-0.9072	0.04054	-0.8499	-0.1674	3.53	2.35	0.4697
13	209608_s_at	39	ACAT2	acetyl-CoA a	1.28	0.04431	-0.6576	15.32	0.06943	-0.4848	-1.33	-1.76	-2.89	0.144	-0.003197
14	202324_s_at	64746	ACBD3	acyl-CoA bin	0.006175	-1.59	-0.4238	-0.05171	0.7066	0.957	0.5078	0.8753	0.9008	1.16	-0.6768
15	204617_s_at	65057	ACD	adrenocortic	0.7791	0.7324	-1.48	0.4226	11.94	-0.8241	-1.42	-0.2702	-3.28	-1.7	0.5791
16	210337_s_at	47	ACLY	ATP citrate ly	0.2562	-0.9472	0.4287	-0.7229	0.7186	0.07974	0.7812	1.26	0.204	0.5701	0.863
17	221641_s_at	23597	ACOT9	acyl-CoA thio	0.5095	-2.08	0.05166	-1.57	0.1091	-0.5048	-1.65	-0.002708	-1.52	0.06977	0.6687
18	202604_x_at	102	ADAM10	ADAM metall	-1.8	-1.44	0.859	0.4998	1.01	0.6541	2.03	1.15	0.8074	1.81	1.01
19	219384_s_at	23536	ADAT1	adenosine de	-0.1221	1.57	0.5826	1.69	-0.3039	-0.01727	-0.001226	-0.1983	-0.4639	-0.8803	0.9154
20	218168_s_at	56997	ADCK3	aarF domain	6.79	-0.4081	-2.26	5.94	1.1	0.4086	7.38	0.4925	0.07287	-0.5016	-1.4
21	208847_s_at	128	ADHS	alcohol dehyd	1.16	-1.85	-0.6653	-3.87	-0.3007	-0.09385	-1.1	0.0344	-0.2797	-0.7684	-0.7147
22	217761_at	55256	ADI1	acireductone	0.7441	0.01113	0.4289	-2.33	-0.9985	-0.7723	-4.29	-3.52	-0.04159	0.4366	-1.8
23	212500_at	84890	ADO	2-aminoethar	0.5321	-0.4698	0.7953	-0.9001	-0.009903	0.2115	-3.04	-1.37	4.88	-2.52	-0.5321
24	206170_at	154	ADRB2	adrenergic, b	2.45	3.88	4.12	3.22	0.9169	-0.07595	0.6957	-0.2924	7.95	-0.4622	3.59
25	203566_s_at	178	ADG1	amula, alpha	0.8671	1.59	0.2876	0.6563	-0.6986	-0.999	-1.76	-1.56	-0.1979	1.09	-0.3012

Figure 9-110mb.csv

AutoSave

1.3gb.csv

Search

TEJA KUMAR REDDY KOVURI

FileHomeInsertPage LayoutFormulasDataReviewViewHelp

ClipboardFontAlignmentNumberStylesCellsEditingAnalysisSensitivity

ShareComments

POSSIBLE DATA LOSS

Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format.

Don't show againSave As...

A1Emp ID

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Emp ID	Name Pre	First Name	Middle Initial	Last Name	Gender	E Mail	Father's Name	Mother's Name	Mother's Initial	Date of Birth	Time of Birth	Age in Yrs	Weight in	Date of Join	Quarter of	Half of Join	Year of Join	Month of	Month Name	Short
2	742048	Drs.	Lizeth	P	Mccoll	F	lizeth.mcc	Renato M	Serena M	Broxton	10/15/196	47.19	49	12/16/200	Q1	H1	1995	1	January	Jan	
3	671135	Ms.	Argentina	O	Hern	F	argentina.h	Earl Hern	Chrissy H	Tapley	10/15/196	57.92	53	12/16/200	Q2	H1	1986	4	April	Apr	
4	965851	Mr.	Damian	N	Patillo	M	damian.pi	Harley Pat	Lucinda P	Etter	10/15/196	45.51	84	12/16/200	Q4	H2	2004	12	December	Dec	
5	224660	Drs.	Imogene	P	Hagopian	F	imogene.i	Delmar H	Carolina H	Lockett	2/19/1995	25.55	52	12/16/200	Q2	H1	2017	6	June	Jun	
6	852694	Mr.	Walker	E	Wallach	M	walker.w	Gale Wall	Georgian	Creekmor	10/15/196	28.09	50	2/28/2020	Q1	H1	2020	2	February	Feb	
7	144102	Ms.	Jesusa	W	Hollie	F	jesusita.h	Clement	Nilda Holl	Barone	10/30/198	32.86	49	12/16/200	Q1	H1	2015	3	March	Mar	
8	687793	Mr.	Fausto	P	Esqueda	M	fausto.es	Randal Es	Yang Esqu	Cullens	4/23/1971	49.39	75	10/29/199	Q4	H2	1998	10	October	Oct	
9	636308	Mrs.	Vanda	S	Komar	F	vanda.kor	Carlo Kom	Katia Kom	Almanzar	10/15/196	58.04	52	7/25/2018	Q3	H2	2018	7	July	Jul	
10	218660	Mrs.	Destiny	A	Nicholson	F	destiny.ni	Ramon Ni	Tanika Ni	Schauer	10/15/196	25.19	54	11/22/201	Q4	H2	2016	11	November	Nov	
11	465691	Ms.	Evie	V	Hamby	F	evie.ham	Devin Han	Jericka Han	Plum	6/22/1980	40.22	60	12/16/200	Q4	H2	2013	10	October	Oct	
12	242339	Dr.	Carlton	G	Friedrich	M	carlton.fri	Tommy Fr	Ricki Fric	Camp	10/15/196	26.92	64	12/16/200	Q4	H2	2016	10	October	Oct	
13	636949	Mr.	Carmelo	A	Haynie	M	carmelo.h	Mason Ha	Dorian Ha	Blessing	3/19/1988	32.47	83	12/16/200	Q1	H1	2015	3	March	Mar	
14	890565	Ms.	Bari	K	Troche	F	bari.troch	Wilson Tr	Madison T	Imhoff	10/15/196	59.26	40	12/16/200	Q2	H1	2006	6	June	Jun	
15	206270	Mr.	Burton	B	Lowry	M	burton.b	Lowri Oren	Fredda Lo	Goodwyn	9/18/1987	32.98	50	10/29/201	Q4	H2	2011	10	October	Oct	
16	418390	Prof.	Mickey	Z	Bassham	M	mickey.ba	Deon Bass	Lan Bassh	Buttler	9/29/1997	22.94	57	12/19/201	Q4	H2	2019	12	December	Dec	
17	359884	Ms.	Ying	Y	Cerrone	F	ying.cerro	Benedict	Jeanett C	Line	3/19/1980	40.48	53	5/17/2012	Q2	H1	2012	5	May	May	
18	301840	Mrs.	Bette	O	Copas	F	bette.cop	Rupert Co	Bette Cop	Gerner	10/15/196	57.84	55	2/26/2000	Q1	H1	2000	2	February	Feb	
19	232212	Mrs.	Tien	W	Harkey	F	tien.harke	Robin Har	Carlotta H	Blas	10/15/196	22.01	52	5/26/2020	Q2	H1	2020	5	May	May	
20	450156	Ms.	Loma	S	Chesney	F	loma.ches	Bret Ches	Viola Che	Barnhill	5/25/1963	57.31	50	11/14/198	Q4	H2	1989	11	November	Nov	
21	158031	Mr.	Ruben	K	Pastor	M	ruben.pas	Filiberto	F Dona Past	Audette	11/16/199	26.81	83	11/20/201	Q4	H2	2016	11	November	Nov	

1.3gb

Ready

100%

Figure 10-1.3gb.csv

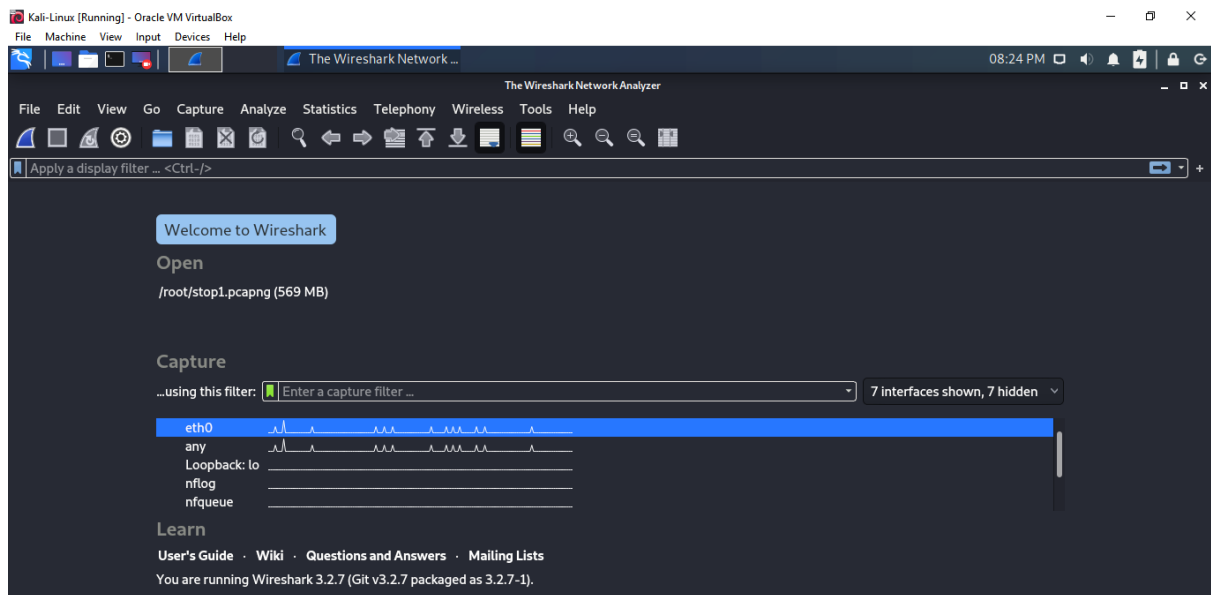


Figure 11-Wireshark in Kali Linux

0030	01	fe	3b	df	00	00	01	01	08	0a	8a	c6	98	a1	c0	41		A
0040	6d	2b	44	53	28	70	68	29	44	45	44	44	49	53	4b	5f	m+DS(ph)	DEDDISK
0050	15	00	00	41	41	44	45	4c	47	49	44	56	44	53	28	31	·A	ADEL GIDVDS(1
0060	29	44	45	44	44	49	53	4b	23	00	00	41	41	44	45	4c)	DEDDISK #·ADEL
0070	47	49	44	56	44	53	28	31	34	39	2e	32	39	29	44	45	GIDVDS((1 49.29) DE
0080	44	44	49	53	28	2d	31	34	39	2e	32	39	29	4b	07	00	DIS(-14	9.29)K·
0090	00	59	42	52	31	34	32	57	3f	00	00	41	54	50	2d	64	YBR142W ?·	ATP-d
00a0	65	70	65	6e	64	65	6e	74	20	52	4e	41	20	68	65	6c	e	pendent RNA hel
00b0	69	63	61	73	65	20	4d	41	4b	35	3b	4d	61	69	6e	74	icase MA K5;Maint	
00c0	65	6e	61	6e	63	65	20	6f	66	20	6b	69	6c	6c	65	72	enance o f killer	
00d0	20	70	72	6f	74	65	69	6e	20	35	06	00	00	50	33	38	protein 5·P38	
00e0	31	31	32	23	00	00	41	41	44	45	4c	47	49	44	56	44	112#·AA	DELGIDVD
00f0	53	28	31	39	33	2e	30	35	29	44	45	44	44	49	53	28	S(193.05)	DEDDIS(
0100	2d	31	39	33	2e	30	35	29	4b	1f	00	00	41	41	44	51	-193.05)	K·AADQ
0110	41	44	45	53	53	50	4c	4c	53	50	53	4e	53	4e	48	50	ADESSPLL	SPSNSNHP

Figure 12-Data visibility in Wireshark

Z	AA	AB	AC	AD	AE	AF	AG
<i>Mascot Score</i>	<i>Mascot Delta</i>	<i>Mascot Delta Other Pep.</i>	<i>PTM Score</i>	<i>PTM Delta</i>	<i>Ratio H/L</i>	<i>Ratio H/L Normalized</i>	<i>Intensity</i>
50.1	50.1	50.1	166.7	166.7	0.88553	1.0539	88908
38.37	38.37	38.37	82.838	44.076	0.81788	1.0287	27794
73.08	73.08	73.08	279.78	149.29	1.0034	1.234	90966
89.6	89.6	89.6	356.47	193.05	0.97906	1.1535	334570
8.54	8.54	8.54	24.663	3.991	0.70302	0.88731	23193
9.52	9.52	9.52	33.838	4.7778	0.71957	0.87467	34266
4.77	4.77	4.77	28.626	0	0.42436	0.52958	19730
14.04	14.04	14.04	49.581	5.7417	0.70228	0.84442	47539
38.1	38.1	38.1	74.643	74.643	0.90634	1.1439	23220
45.96	45.96	45.96	66.706	66.706	0.94331	1.2075	14025

Figure 13-Actual data in Excel File

```

01b0 61 79 2c 54 75 65 2c 33 2e 37 38 2c 31 32 36 30 ay,Tue,3 .78,1260
01c0 34 38 2c 31 39 25 2c 33 38 36 2d 33 37 2d 35 36 48,19%,3 86-37-56
01d0 32 38 2c 32 30 31 2d 39 39 37 2d 33 39 35 37 2c 28,201-9 97-3957,
01e0 41 74 6c 61 6e 74 69 63 20 43 69 74 79 2c 41 74 Atlantic City,At
01f0 6c 61 6e 74 69 63 2c 41 74 6c 61 6e 74 69 63 20 lantic,A tantic
0200 43 69 74 79 2c 4e 4a 2c 38 34 30 31 2c 4e 6f 72 City,NJ, 8401,Nor
0210 74 68 65 61 73 74 2c 64 61 6e 69 63 68 6f 6c 73 theast,d anichols
0220 6f 6e 2c 73 7c 3f 74 2f 25 58 2b 0d 0a 34 36 35 on,s|?t/ %X+..465
0230 36 39 31 2c 4d 73 2e 2c 45 76 69 65 2c 56 2c 48 691,Ms Evie,V,H
0240 61 6d 62 79 2c 46 2c 65 76 69 65 2e 68 61 6d 62 amby,F e vie.hamb
0250 79 40 67 6d 61 69 6c 2e 63 6f 6d 2c 44 65 76 69 y@gmail. com Devi
0260 6e 20 48 61 6d 62 79 2c 4a 65 72 69 63 61 20 48 n Hamby, Jerica H
0270 61 6d 62 79 2c 50 6c 75 6d 2c 36 2f 32 32 2f 31 amby,Plu m,6/22/1
0280 39 38 30 2c 31 32 3a 31 37 3a 34 32 20 41 4d 2c 980,12:1 7:42 AM,
0290 34 30 2e 32 32 2c 36 30 2c 31 40.22,60 ,1

```

Figure 14-data identified - evie.hamby@gmail.com

9	636308	Mrs.	Vanda	S	Komar	F	vanda.komar@aol.com	Carlo Kom	Katia
10	218660	Mrs.	Destiny	A	Nicholson	F	destiny.nicholson@gmail.com	Ramon Ni	Tanika
11	465691	Ms.	Evie	V	Hamby	F	evie.hamby@gmail.com	Devin Han	Jerica
12	242339	Dr.	Carlton	G	Friedrich	M	carlton.friedrich@gmail.com	Tommy Fr	Ricki F

Figure 15-Actual Data in Excel File - evie.hamby@gmail.com

```

0030 01 fe 70 9a 00 00 01 01 08 0a 8b 1b 7d af c0 96 p
0040 52 29 32 2e 37 36 34 34 38 32 34 38 38 34 0d 0a R)2.7644 824884..
0050 22 53 6f 75 74 68 22 2c 22 50 72 6f 64 20 53 22 "South", "Prod S"
0060 2c 32 30 31 32 2d 30 33 2d 30 32 2c 34 34 30 30 .2012-03 -02 4400
0070 35 2e 33 38 31 39 33 32 34 33 37 39 0d 0a 22 53 5.381932 4379..S
0080 6f 75 74 68 22 2c 22 50 72 6f 64 20 4b 22 2c 32 outh", "P rod K", 2
0090 30 31 31 2d 30 38 2d 31 32 2c 33 34 30 39 37 2e 011-08-1 2,34097.
00a0 32 34 34 38 30 35 31 36 32 34 0d 0a 22 53 6f 75 24480516 24..Sou
00b0 74 68 22 2c 22 50 72 6f 64 20 50 22 2c 32 30 31 th", "Pro d P", 201
00c0 30 2d 30 36 2d 31 39 2c 31 33 31 32 37 2e 39 32 0-06-19, 13127.92
00d0 37 31 36 33 36 35 38 0d 0a 22 4e 6f 72 74 68 22 7163658.. "North"
00e0 2c 22 50 72 6f 64 20 41 22 2c 32 30 31 33 2d 30 , "Prod A ", 2013-0
00f0 35 2d 30 34 2c 36 35 34 36 38 2e 32 36 31 38 33 5-04,654 68.26183
0100 37 38 34 36 39 0d 0a 22 4e 6f 72 74 68 22 2c 22 78469.. " North", "
0110 50 72 6f 64 20 41 22 2c 32 30 31 35 2d 30 32 2d Prod A", 2015-02-

```

Figure 16-Data identified in Wireshark-44005

97678	South	Prod B	#####	103429.7
97679	West	Prod K	#####	5908.56
97680	North	Prod T	#####	44005.63
97681	North	Prod C	#####	63105.52

Figure 17-Actual Data in Excel File-44005

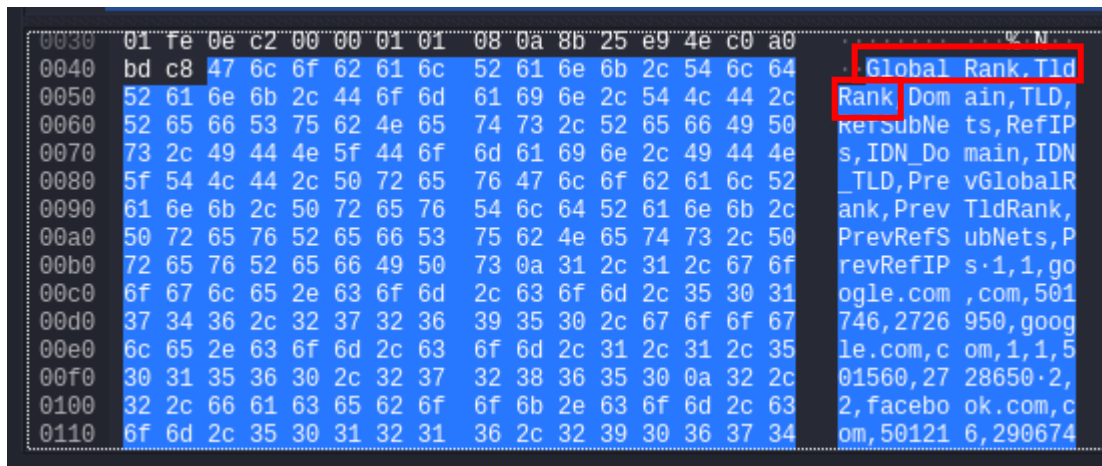


Figure 18-Data identified in Wireshark

	A	B	C	D	E	F	G	H	I	J	K	L
1	GlobalRank	TldRank	Domain	TLD	RefSubNe	RefIPs	IDN_Dom	IDN_TLD	PrevGloba	PrevTldRa	PrevRefSu	PrevRefIPs
2	1	1	google.co	com	501746	2726950	google.co	com	1	1	501560	2728650

Figure 19-Actual data in Excel Data

```

ktkr@ktkr-VirtualBox:~/Desktop/tcp-filetransfer$ ipfs add 20mb.xls
added QmZ6LnMDVoRabqXZ6ZNH7r651GCTdXWbjyDshi8wVL4au3 20mb.xls
20.35 MiB / 20.35 MiB [=====] 100.00%

```

Figure 20-Peer-to-Peer Model Server Machine for 20mb.xls file

```

ktkr@ktkr-VirtualBox:~/Desktop/tcp-filetransfer/ipfs$ ipfs get QmZ6LnMDVoRabqXZ6ZNH7r651GCTdXWbjyDshi8wVL4au3
Saving file(s) to QmZ6LnMDVoRabqXZ6ZNH7r651GCTdXWbjyDshi8wVL4au3
20.35 MiB / 20.35 MiB [=====] 100.00% 35

```

Figure 21-Peer-to-Peer Model Client Machine -2 using IPFS Protocol

192.168.56.105	192.168.56.104	TCP	254 4001 → 4001 [PSH, ACK] Seq=40
192.168.56.103	192.168.56.105	TCP	66 4001 → 4001 [ACK] Seq=2138663
192.168.56.104	192.168.56.105	TCP	108 4001 → 4001 [PSH, ACK] Seq=28
192.168.56.103	192.168.56.105	TCP	107 4001 → 4001 [PSH, ACK] Seq=21
192.168.56.103	192.168.56.105	TCP	107 4001 → 4001 [PSH, ACK] Seq=21
192.168.56.103	192.168.56.105	TCP	107 4001 → 4001 [PSH, ACK] Seq=21
192.168.56.105	192.168.56.104	TCP	87 4001 → 4001 [PSH, ACK] Seq=40
192.168.56.104	192.168.56.105	TCP	87 4001 → 4001 [PSH, ACK] Seq=28

Figure 22-Traffic Analysis in Wireshark for 20mb.xls file

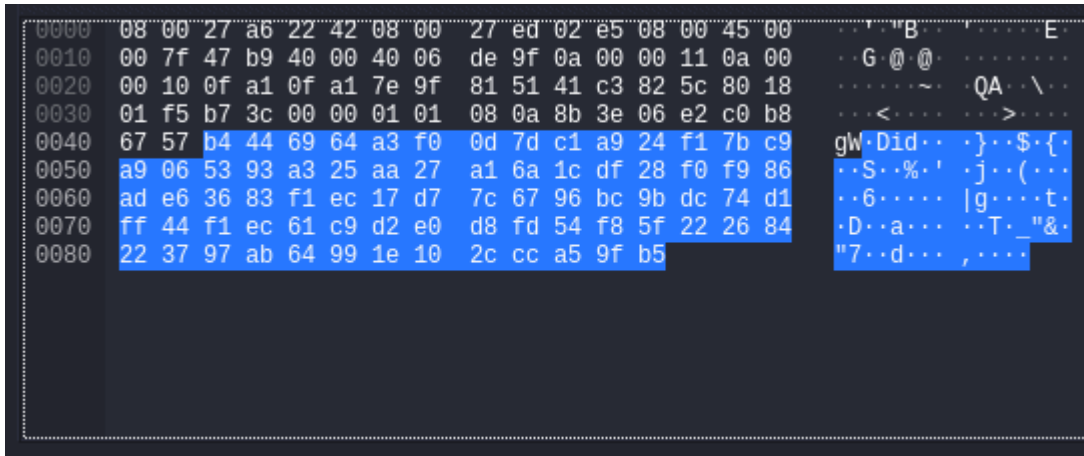


Figure 23-Encrypted data for 20mb.xls file in Wireshark

```
ktkr@ktkr-VirtualBox: ~/Desktop/tcp-filetransfer$ ipfs add 1.3gb.csv
added QmPwHhvmAAx8P3pjJVKRq3eyjUv7q7fqWvvepcrPo2ZsAP 1.3gb.csv
1.24 GiB / 1.24 GiB [=====] 100.00%
```

Figure 24-Peer-to-Peer Model Server Machine using IPFS Protocol for 1.3gb.csv file

```
ktkr@ktkr-VirtualBox: ~/Desktop/tcp-filetransfer/ipfs$ ipfs get QmPwHhvmAAx8P3pjJVKRq3eyjUv7q7fqWvvepcrPo2ZsAP
Saving file(s) to QmPwHhvmAAx8P3pjJVKRq3eyjUv7q7fqWvvepcrPo2ZsAP
1.24 GiB / 1.24 GiB [=====] 100.00% 6m7s
```

Figure 25-Peer-to-Peer Model Client Machine-1 using IPFS Protocol for 1.3gb.csv file

192.168.56.104	192.168.56.103	TCP	108	4001 → 4001	[PSH, ACK]
192.168.56.103	192.168.56.104	TCP	66	4001 → 4001	[ACK] Seq=2
192.168.56.103	192.168.56.105	TCP	174	4001 → 4001	[PSH, ACK]
192.168.56.103	192.168.56.104	TCP	174	4001 → 4001	[PSH, ACK]
192.168.56.104	192.168.56.103	TCP	66	4001 → 4001	[ACK] Seq=1
192.168.56.105	192.168.56.103	TCP	109	4001 → 4001	[PSH, ACK]
192.168.56.103	192.168.56.105	TCP	66	4001 → 4001	[ACK] Seq=1
192.168.56.104	192.168.56.103	TCP	109	4001 → 4001	[PSH, ACK]
192.168.56.104	192.168.56.103	TCP	177	4001 → 4001	[PSH, ACK]
192.168.56.103	192.168.56.104	TCP	66	4001 → 4001	[ACK] Seq=3
192.168.56.105	192.168.56.103	TCP	177	4001 → 4001	[PSH, ACK]

Figure 26-Traffic in Wireshark

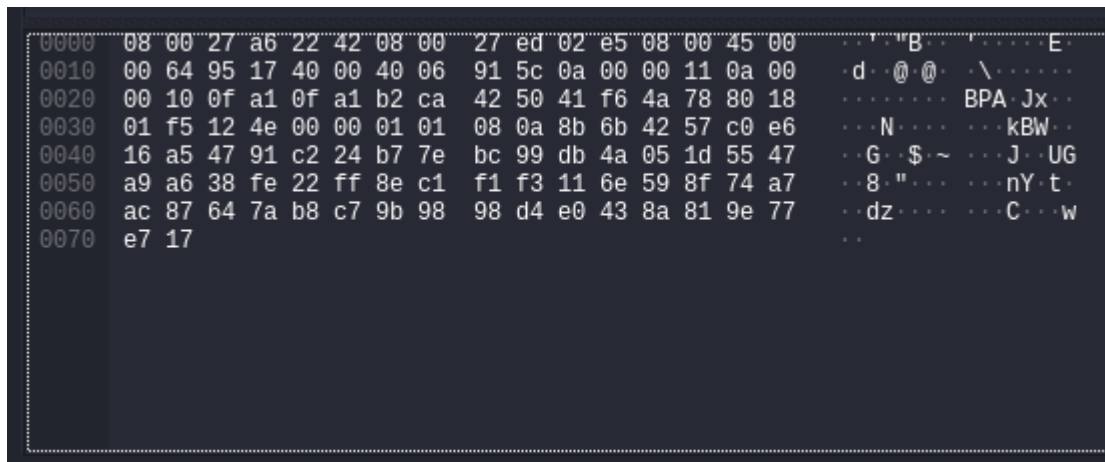


Figure 27-Encrypted Data in Wireshark

```
ktkr@ktkr-VirtualBox: ~/Desktop/tcp-filetransfer$ ipfs add 110mb.csv
added QmTBCQKJJ5lbbc18hujtES3PtT7Q5gyYfGjqaxoyjCd3Zb 110mb.csv
108.81 MiB / 108.81 MiB [=====] 100.00%
```

Figure 28-Peer-to-Peer Model Server Machine using IPFS Protocol for 110mb.csv file

```
ktkr@ktkr-VirtualBox: ~/Desktop/tcp-filetransfer/ipfs$ ipfs get QmTBCQKJJ5lbbc18hujtES3PtT7Q5gyYfGjqaxoyjCd3Zb
Saving file(s) to QmTBCQKJJ5lbbc18hujtES3PtT7Q5gyYfGjqaxoyjCd3Zb
108.81 MiB / 108.81 MiB [=====] 100.00% 24s
```

Figure 29-Peer-to-Peer Model Client Machine-2 using IPFS Protocol for 110mb.csv file

192.168.56.103	192.168.56.104	TCP	108 4001 → 4001 [PSH, ACK] Seq=1163756
192.168.56.105	192.168.56.104	TCP	87 4001 → 4001 [PSH, ACK] Seq=155514
192.168.56.103	192.168.56.104	TCP	176 4001 → 4001 [PSH, ACK] Seq=1163756
192.168.56.104	192.168.56.105	TCP	91 4001 → 4001 [PSH, ACK] Seq=218211
192.168.56.104	192.168.56.105	TCP	254 4001 → 4001 [PSH, ACK] Seq=218236
192.168.56.105	192.168.56.104	TCP	107 4001 → 4001 [PSH, ACK] Seq=155535

Figure 30-Traffic in Wireshark

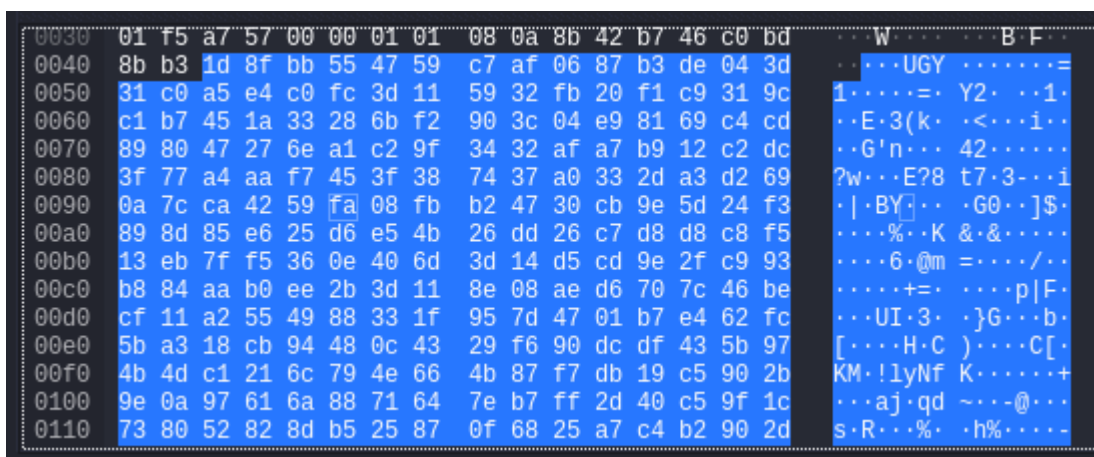


Figure 31-Encrypted Data in Wireshark

```
ktkr@ktkr-VirtualBox:~/Desktop/tcp-filetransfer$ ipfs add 70mb.csv
added QmP1fnLScFNmDXJbcwYyDBJks2surdVZUszHWMXQXVb2U 70mb.csv
77.33 MiB / 77.33 MiB [=====] 100.00%
```

Figure 32-Peer-to-Peer Model Server Machine using IPFS Protocol for 70mb.csv file

```
ubuntu@ubuntu-VirtualBox:~/Desktop/tcp-filetransfer$ ipfs get QmP1fnLScFNmDXJbcwYyDBJks2surdVZUszHWMXQXVb2U
Saving file(s) to QmP1fnLScFNmDXJbcwYyDBJks2surdVZUszHWMXQXVb2U
77.35 MiB / 77.35 MiB [=====] 100.00% 16s
ubuntu@ubuntu-VirtualBox:~/Desktop/tcp-filetransfer$
```

Figure 33-Peer-to-Peer Model Client Machine-1 using IPFS Protocol for 70mb.csv file

192.168.56.103	192.168.56.105	TCP	107	4001 → 4001	[PSH, ACK]	Seq=81399982
192.168.56.105	192.168.56.104	TCP	169	4001 → 4001	[PSH, ACK]	Seq=153850 Ac
192.168.56.105	192.168.56.103	TCP	254	4001 → 4001	[PSH, ACK]	Seq=287025 Ac
192.168.56.104	192.168.56.105	TCP	108	4001 → 4001	[PSH, ACK]	Seq=110506 Ac
192.168.56.104	192.168.56.105	TCP	176	4001 → 4001	[PSH, ACK]	Seq=110548 Ac
192.168.56.105	192.168.56.103	TCP	87	4001 → 4001	[PSH, ACK]	Seq=287213 Ac
192.168.56.105	192.168.56.104	TCP	66	4001 → 4001	[ACK]	Seq=153953 Ack=110
192.168.56.103	192.168.56.105	TCP	108	4001 → 4001	[PSH, ACK]	Seq=81400023
192.168.56.103	192.168.56.105	TCP	87	4001 → 4001	[PSH, ACK]	Seq=81400065

Figure 34-Traffic in Wireshark for 70mb.csv file

0030	01 f5 2a e7 00 00 01 01 08 0a 8b 49 32 54 c0 c4I2T
0040	06 c7 cc a2 d4 e4 a1 d2 00 fd f9 bf 86 b3 5a 98Z
0050	06 a3 4e 0e ec 30 59 a4 2e f1 f1 5b 0b f7 5b 5a	..N..0Y..[..Z
0060	27 29 b7 1e 6d 71 1a 6f 27 db 0c dc 3f 6c 7c 17	..mq..o'..?1 .
0070	55 4d 41 ce a5 44 cd 78 4e 37 84 37 46 1e 50 fd	UMA..D.x N7.7F.P.
0080	b7 cd b6 d4 13 95 95 73 8b 7d 55 3a c2 b1 be 6es..}U:..n
0090	4a 50 fa 23 a8 d6 bc 83 20 61 80 5e 91 5b 66 cf	JP.#....a.^.[f.
00a0	12 1f ce 0d 62 ef 41 e4 2d 14 50 37 6b 11 80 a0b.A..-P7k...
00b0	20 10 a7 bf fc 11 3d e7 3e 60 aa e2 44 0f 40 36=>...D.@6
00c0	b4 0f 6f 80 88 43 85 50 c5 f3 65 f6 c2 fc 90 9e	...o..C.P..e....
00d0	b8 83 02 36 b3 6e bb 01 3b d0 1e e7 fa 44 73 d7	...6.n..;....Ds.
00e0	38 ba 7d e1 00 fe bc 47 ba d9 7f 7f 2a 31 e5 eb	8..}....G....*1..
00f0	ef e4 00 20 2b 9d 6f a4 48 f1 04 82 f2 fd f0 33	...+..o..H.....3
0100	af 20 de 97 f6 a0 c4 df aa 2b 40 6b 4b 7a 75 fe+@kKzu.
0110	1f bd 88 91 52 bf d6 81 49 78 bb 61 c8 86 28 e7	...R...Ix.a..(.

Figure 35-Encrypted data in Wireshark