

# Interplanetary File System Assignment

Gokul G (CB.EN.P2CYS20017)

1. Please add the files from the below link into IPFS and provide the hash value

PDF -

[https://intranet.cb.amrita.edu/sites/default/files/M.Tech\\_2021%20Mid%20erm%20RevisedTT.pdf](https://intranet.cb.amrita.edu/sites/default/files/M.Tech_2021%20Mid%20erm%20RevisedTT.pdf)

Image - [https://intranet.cb.amrita.edu/sites/default/files/logo\\_1.png](https://intranet.cb.amrita.edu/sites/default/files/logo_1.png)

Executable - [https://github.com/ipfs-shipyard/ipfs-](https://github.com/ipfs-shipyard/ipfs-desktop/releases/download/v0.13.2/IPFS-Desktop-Setup-0.13.2.exe)

[desktop/releases/download/v0.13.2/IPFS-Desktop-Setup-0.13.2.exe](https://github.com/ipfs-shipyard/ipfs-desktop/releases/download/v0.13.2/IPFS-Desktop-Setup-0.13.2.exe)

Ans)

CID of pdf is: QmXsXwqkX9g2trXAWvGy8yH5W6ktfLhbfjkf1UztRL8cXy

```
gokul ~/Documents/mtech-3sem/blockchain/assign02
$ wget -q https://intranet.cb.amrita.edu/sites/default/files/M.Tech_2021%20Mid%20erm%20RevisedTT.pdf
gokul ~/Documents/mtech-3sem/blockchain/assign02
$ ipfs add M.Tech_2021\ Mid\ erm\ RevisedTT.pdf
added QmXsXwqkX9g2trXAWvGy8yH5W6ktfLhbfjkf1UztRL8cXy M.Tech_2021 Mid erm_ RevisedTT.pdf
1.65 MiB / 1.65 MiB [=====] 100.00%
gokul ~/Documents/mtech-3sem/blockchain/assign02
$
```

CID of image is: QmWpUpjVmYHiaHyD7mELN6jfe5hn2NguByyY3wLszw3yfG

```
gokul ~/Documents/mtech-3sem/blockchain/assign02
$ wget -q https://intranet.cb.amrita.edu/sites/default/files/logo_1.png
gokul ~/Documents/mtech-3sem/blockchain/assign02
$ ipfs add logo_1.png
added QmWpUpjVmYHiaHyD7mELN6jfe5hn2NguByyY3wLszw3yfG logo_1.png
14.55 KiB / 14.55 KiB [=====] 100.00%
gokul ~/Documents/mtech-3sem/blockchain/assign02
$
```

CID of exe is: QmTzNqrxYghhA55xNJsvLRNZ1DQzoyeUdkjiLV4onH1qHy

```
gokul ~/Documents/mtech-3sem/blockchain/assign02
$ wget -q https://github.com/ipfs-shipyard/ipfs-desktop/releases/download/v0.13.2/IPFS-Desktop-Setup-0.13.2.exe
gokul ~/Documents/mtech-3sem/blockchain/assign02
$ ipfs add IPFS-Desktop-Setup-0.13.2.exe
added QmTzNqrxYghhA55xNJsvLRNZ1DQzoyeUdkjiLV4onH1qHy IPFS-Desktop-Setup-0.13.2.exe
81.78 MiB / 81.78 MiB [=====] 100.00%
gokul ~/Documents/mtech-3sem/blockchain/assign02
$
```

2. What is the content of the below file

"Qmbv52Ft9ybXMdTkS4usQnC85qjwCi4RkmFhxhc7nJg8bT"

Ans) File not available

### 3. Difference between CIDv0 and CIDv1?

Ans)

Parameter	CID v0	CID v1
Encoding scheme	Base58btc	base32 as default, can be changed
Way to identify	Starts with Qm	Starts with ba for base32
Hash algorithm used	SHA2-256	Multiple algorithms supported
Representation	HashID_Hashlength_ Digest	MultiBase_CIDversion_MultiCodec_HashID_ _hashLength_Digest

### 4. Analyse the below IPFS CIDs and classify them.

QmbWqxBEKC3P8tqsKc98xmWNzrzDtRLMiMPL8wBuTGsMnR

bafybeigdyrzt5sfp7udm7hu76uh7y26nf3efuylqabf3oclgty55fbzdi

QmT78zSuBmuS4z925WZfrqQ1qHaJ56DQaTfyMUF7F8ff5o

cafyyqqigdyrzt5sfp7udm7au76uh7y26nf3efuylqabf4oclgty44fbzdi

Ans)

Data pulled from [IPFS CID inspector](#)

a) QmbWqxBEKC3P8tqsKc98xmWNzrzDtRLMiMPL8wBuTGsMnR

```
CID Docs Spec Tutorial
QmbWqxBEKC3P8tqsKc98xmWNzrzDtRLMiMPL8wBuTGsMnR

HUMAN READABLE CID
base58btc - cidv0 - dag-pb - (sha2-256 : 256 : C3C4733EC8AFFD06CF9E9FF50FFC6BCD2EC85A6170004BB709669C31DE94391A)
MULTIBASE - VERSION - MULTICODEC - MULTIHASH (NAME : SIZE : DIGEST IN HEX)

MULTIBASE
PREFIX:
implicit
NAME:
base58btc

MULTICODEC
CODE:
implicit
NAME:
dag-pb

MULTIHASH
CODE:
0x12
NAME:
sha2-256
BITS:
256
DIGEST (HEX):
C3C4733EC8AFFD06CF9E9FF50FFC6BCD2EC85A6170004BB709669C31DE94391A

CIDV1 (BASE32)
bafybeigdyrzt5sfp7udm7hu76uh7y26nf3efuylqabf3oclgty55fbzdi
```

b) bafybeigdyrzt5sfp7udm7hu76uh7y26nf3efuylqabf3oclgqty55fbzdi

CID [Docs](#) [Spec](#) [Tutorial](#)

bafybeigdyrzt5sfp7udm7hu76uh7y26nf3efuylqabf3oclgqty55fbzdi

**HUMAN READABLE CID**  
base32 - cidv1 - dag-pb - (sha2-256 : 256 : C3C4733EC8AFFD06CF9E9FF50FFC6BCD2EC85A6170004BB709669C31DE94391A)  
MULTIBASE - VERSION - MULTICODEC - MULTIHASH (NAME : SIZE : DIGEST IN HEX)

**MULTIBASE**  
PREFIX:  
b  
NAME:  
base32

**MULTICODEC**  
CODE:  
0x70  
NAME:  
dag-pb

**MULTIHASH**  
CODE:  
0x12  
NAME:  
sha2-256  
BITS:  
256  
DIGEST (HEX):  
C3C4733EC8AFFD06CF9E9FF50FFC6BCD2EC85A6170004BB709669C31DE94391A

**CIDV1 (BASE32)**  
bafybeigdyrzt5sfp7udm7hu76uh7y26nf3efuylqabf3oclgqty55fbzdi

c) QmT78zSuBmuS4z925WZfrqQ1qHaJ56DQaTfyMUF7F8ff5o

CID [Docs](#) [Spec](#) [Tutorial](#)

QmT78zSuBmuS4z925WZfrqQ1qHaJ56DQaTfyMUF7F8ff5o

**HUMAN READABLE CID**  
base58btc - cidv0 - dag-pb - (sha2-256 : 256 : 46D44814B9C5AF141C3AAAB7C05DC5E844EAD5F91F12858B021EBA45768B4C0E)  
MULTIBASE - VERSION - MULTICODEC - MULTIHASH (NAME : SIZE : DIGEST IN HEX)

**MULTIBASE**  
PREFIX:  
implicit  
NAME:  
base58btc

**MULTICODEC**  
CODE:  
implicit  
NAME:  
dag-pb

**MULTIHASH**  
CODE:  
0x12  
NAME:  
sha2-256  
BITS:  
256  
DIGEST (HEX):  
46D44814B9C5AF141C3AAAB7C05DC5E844EAD5F91F12858B021EBA45768B4C0E

**CIDV1 (BASE32)**  
bafybeicg2rebjoofv4kbyovkw7af3rpiitvnl6i7ckcywaq6xjcxnc2mby

d) `cafyyqqigdyrzt5sfp7udm7au76uh7y26nf3efuylqabf4ocltqy44fbzdi`

CID [Docs](#) [Spec](#) [Tutorial](#)

`cafyyqqigdyrzt5sfp7udm7au76uh7y26nf3efuylqabf4ocltqy441`

**HUMAN READABLE CID**  
`base32pad - cidv1 - dag-cbor - (undefined : 256 : C3C4733EC8AFFD06CF829FF50FFC6BCD2EC85A6170004BC709669C31CE14391A)`  
MULTIBASE - VERSION - MULTICODEC - MULTIHASH (NAME : SIZE : DIGEST IN HEX)

**MULTIBASE**  
PREFIX:  
`c`  
NAME:  
`base32pad`

**MULTICODEC**  
CODE:  
`0x71`  
NAME:  
`dag-cbor`

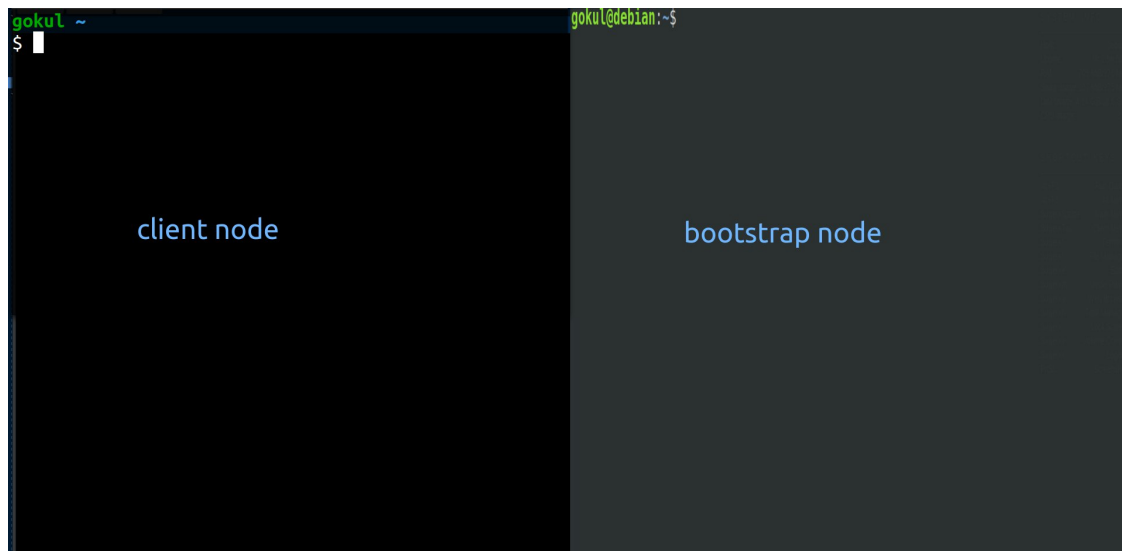
**MULTIHASH**  
CODE:  
`0x08`  
NAME:  
`undefined`  
BITS:  
`256`  
DIGEST (HEX):  
`C3C4733EC8AFFD06CF829FF50FFC6BCD2EC85A6170004BC709669C31CE14391A`

**CIDV1 (BASE32)**  
`bafyyqqigdyrzt5sfp7udm7au76uh7y26nf3efuylqabf4ocltqy44fbzdi`

**5. How to configure a Private IPFS, please list down the steps with screen shots.**

**Ans)**

Minimum two nodes are required, one acting as bootstrap node and one acting as client node. Two VMs are used for this purpose.



step1: Install the IPFS Go client on both VMs

```
gokul@debian: ~/Desktop/go-ipfs
gokul@debian:~/Desktop$ wget -q https://dist.ipfs.io/go-ipfs/v0.10.0/go-ipfs_v0.10.0_linux-amd64.tar.gz
gokul@debian:~/Desktop$
gokul@debian:~/Desktop$ tar -xvf go-ipfs_v0.10.0_linux-amd64.tar.gz
go-ipfs/LICENSE
go-ipfs/LICENSE-APACHE
go-ipfs/LICENSE-MIT
go-ipfs/README.md
go-ipfs/install.sh
go-ipfs/ipfs
gokul@debian:~/Desktop$ cd go-ipfs/
gokul@debian:~/Desktop/go-ipfs$ chmod u+x i
install.sh ipfs
gokul@debian:~/Desktop/go-ipfs$ chmod u+x install.sh
gokul@debian:~/Desktop/go-ipfs$ sudo ./install.sh
[sudo] password for gokul:
Moved ./ipfs to /usr/local/bin
gokul@debian:~/Desktop/go-ipfs$ ip
ip ipcmk ipcrm ipcs ipfs iptables-xml
gokul@debian:~/Desktop/go-ipfs$ whereis ipfs
ipfs: /usr/local/bin/ipfs
gokul@debian:~/Desktop/go-ipfs$
```

download the tar.gz file

extract files from archive

make the sh program executable

Program copies binary to global path folder

ipfs binary is available globally

step2: Initialize IPFS on both nodes

```

gokul@debian:~$ ipfs init
generating ED25519 keypair...done
peer identity: 12D3KooWQXgW9CX5uXvyv9MGiEAGXLsagQCLMUxLpXJasykkDa32
initializing IPFS node at /home/gokul/.ipfs
to get started, enter:

    ipfs cat /ipfs/QmQPeNsJPYVWPFdVHb77w8G42Fvo15z4bG2X8D2GhfbSXc/readme

gokul@debian:~$ ls -llh ~/.ipfs/
total 24K
drwxr-xr-x 25 gokul gokul 4.0K Dec  4 21:48 blocks
-rw----- 1 gokul gokul 3.7K Dec  4 21:48 config
drwxr-xr-x 2 gokul gokul 4.0K Dec  4 21:48 datastore
-rw----- 1 gokul gokul 190 Dec  4 21:48 datastore_spec
drwx----- 2 gokul gokul 4.0K Dec  4 21:48 keystore
-rw-r--r-- 1 gokul gokul  3 Dec  4 21:48 version
gokul@debian:~$

```

step3: Create a private IPFS network

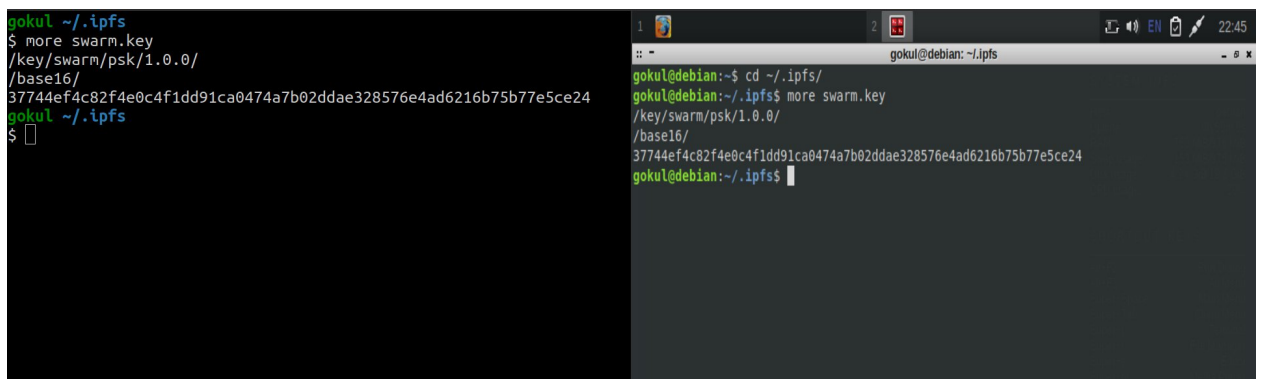
3.1) Create swarm key on bootstrap node

```

gokul@debian:~$ echo -e "/key/swarm/psk/1.0.0/\n/base16/\n`tr -dc 'a-f0-9'< /dev
/urandom | head -c64`" > ~/.ipfs/swarm.key
gokul@debian:~$
gokul@debian:~$ more ~/.ipfs/swarm.key
/key/swarm/psk/1.0.0/
/base16/
37744ef4c82f4e0c4f1dd91ca0474a7b02ddae328576e4ad6216b75b77e5ce24
gokul@debian:~$

```

3.2) Copy swarm key to client node



The image shows two terminal windows. The left window is a terminal on the bootstrap node (gokul@debian) showing the contents of the swarm.key file. The right window is a terminal on the client node (gokul@debian: ~/.ipfs) showing the same contents of the swarm.key file being viewed.

```

gokul ~/.ipfs
$ more swarm.key
/key/swarm/psk/1.0.0/
/base16/
37744ef4c82f4e0c4f1dd91ca0474a7b02ddae328576e4ad6216b75b77e5ce24
gokul ~/.ipfs
$

gokul@debian: ~/.ipfs
gokul@debian: ~/.ipfs$ more swarm.key
/key/swarm/psk/1.0.0/
/base16/
37744ef4c82f4e0c4f1dd91ca0474a7b02ddae328576e4ad6216b75b77e5ce24
gokul@debian: ~/.ipfs$

```

3.3) Remove existing bootstrap config from both nodes



```
gokul ~/.ipfs
$ ipfs bootstrap rm --all
removed /dnsaddr/bootstrap.libp2p.io/p2p/QmNnooDu7bfjPFoTZYxMNLWUQJy
rVwtbZg5gBMjTezGAJN
removed /dnsaddr/bootstrap.libp2p.io/p2p/QmQCU2EcMqAQPR2i9bChDtGNJc
hTbq5TbXJJ16u19uLTa
removed /dnsaddr/bootstrap.libp2p.io/p2p/QmbLHANMoJPWSCR5Zhtx6BHJX9K
iKNN6tpvbUcqanj75Nb
removed /dnsaddr/bootstrap.libp2p.io/p2p/QmcZf59bWwK5XFi76CZX8cbJ4Bh
TzzA3gU1ZjYzCvW3dwt
removed /ip4/104.131.131.82/tcp/4001/p2p/QmaCpDMGvV2BGHeYERUEnRQAwe3
N8SzbUtfsmvsqQLuvuJ
removed /ip4/104.131.131.82/udp/4001/quick/p2p/QmaCpDMGvV2BGHeYERUEnR
QAwe3N8SzbUtfsmvsqQLuvuJ
gokul ~/.ipfs
$

gokul@debian: ~/.ipfs
gokul@debian:~/.ipfs$ ipfs bootstrap rm --all
removed /dnsaddr/bootstrap.libp2p.io/p2p/QmNnooDu7bfjPFoTZYxMNLWUQJy
rVwtbZg5gBMjTezGAJN
removed /dnsaddr/bootstrap.libp2p.io/p2p/QmQCU2EcMqAQPR2i9bChDtGNJc
hTbq5TbXJJ16u19uLTa
removed /dnsaddr/bootstrap.libp2p.io/p2p/QmbLHANMoJPWSCR5Zhtx6BHJX9K
iKNN6tpvbUcqanj75Nb
removed /dnsaddr/bootstrap.libp2p.io/p2p/QmcZf59bWwK5XFi76CZX8cbJ4Bh
TzzA3gU1ZjYzCvW3dwt
removed /ip4/104.131.131.82/tcp/4001/p2p/QmaCpDMGvV2BGHeYERUEnRQAwe3
N8SzbUtfsmvsqQLuvuJ
removed /ip4/104.131.131.82/udp/4001/quick/p2p/QmaCpDMGvV2BGHeYERUEnR
QAwe3N8SzbUtfsmvsqQLuvuJ
gokul@debian:~/.ipfs$
```

### 3.4) Configure bootstrap on both nodes

```
gokul ~/.ipfs
$ ipfs bootstrap add /ip4/192.168.110.255/tcp/4001/ipfs/12D3KooWQXgW
9CX5uXvvyv9MgiEAGXlsagQCLMuxLpXJasykkDa32
added /ip4/192.168.110.255/tcp/4001/ipfs/12D3KooWQXgW9CX5uXvvyv9MgiE
AGXlsagQCLMuxLpXJasykkDa32
gokul ~/.ipfs
$

gokul@debian: ~/.ipfs
gokul@debian:~/.ipfs$ ip a | grep -w "inet"
inet 127.0.0.1/8 scope host lo
inet 192.168.110.45/24 brd 192.168.110.255 scope global dynamic noprefixroute enp0s3
gokul@debian:~/.ipfs$ grep -i "peerid" config
"PeerID": "12D3KooWQXgW9CX5uXvvyv9MgiEAGXlsagQCLMuxLpXJasykkDa32",
gokul@debian:~/.ipfs$ ipfs bootstrap add /ip4/192.168.110.255/tcp/4001/ipfs/12D3KooWQXgW
9CX5uXvvyv9MgiEAGXlsagQCLMuxLpXJasykkDa32
added /ip4/192.168.110.255/tcp/4001/ipfs/12D3KooWQXgW9CX5uXvvyv9MgiEAGXlsagQCLMuxLpXJasyk
kDa32
gokul@debian:~/.ipfs$
```

replace IP address with that of bootstrap node and give peerID of bootstrap node in both machines.

### 4) Start the network

```
gokul ~/ipfs
$ export LIBP2P_FORCE_PNET=1
gokul ~/ipfs
$ ipfs daemon &
[2] 6705
gokul ~/ipfs
$ Initializing daemon...
go-ipfs version: 0.10.0
Repo version: 11
System version: amd64/linux
Golang version: go1.16.8
Swarm is limited to private network of peers with the swarm key
Swarm key fingerprint: 737efe9be70194f45c25cf5b5c392bd1
Swarm listening on /ip4/127.0.0.1/tcp/4001
Swarm listening on /ip4/172.17.0.1/tcp/4001
Swarm listening on /ip4/192.168.110.28/tcp/4001
Swarm listening on /ip6/2409:4072:e8d:b16c:10ef:5ab3:244a:ebd7/tcp/4001
Swarm listening on /ip6/2409:4072:e8d:b16c:f0d8:6bca:7653:7714/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 /ip4/192.168.110.28/tcp/4001
Swarm announcing /ip6/2409:4072:e8d:b16c:f0d8:6bca:7653:7714/tcp/4001
Swarm announcing /ip6:::1/tcp/4001
API server listening on /ip4/127.0.0.1/tcp/5001
WebUI: http://127.0.0.1:5001/webui
Gateway (readonly) server listening on /ip4/127.0.0.1/tcp/8080
Daemon is ready

gokul@debian:~/ipfs$ export LIBP2P_FORCE_PNET=1
gokul@debian:~/ipfs$ ipfs daemon &
[1] 2924
gokul@debian:~/ipfs$ Initializing daemon...
go-ipfs version: 0.10.0
Repo version: 11
System version: amd64/linux
Golang version: go1.16.8
Swarm is limited to private network of peers with the swarm key
Swarm key fingerprint: 737efe9be70194f45c25cf5b5c392bd1
Swarm listening on /ip4/127.0.0.1/tcp/4001
Swarm listening on /ip4/192.168.110.45/tcp/4001
Swarm listening on /ip6/2409:4072:e8d:b16c:a00:27ff:fea8:1145/tcp/4001
Swarm listening on /ip6/2409:4072:e8d:b16c:e660:7f93:23f:8f83/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 /ip4/192.168.110.45/tcp/4001
Swarm announcing /ip6/2409:4072:e8d:b16c:e660:7f93:23f:8f83/tcp/4001
Swarm announcing /ip6:::1/tcp/4001
API server listening on /ip4/127.0.0.1/tcp/5001
WebUI: http://127.0.0.1:5001/webui
Gateway (readonly) server listening on /ip4/127.0.0.1/tcp/8080
Daemon is ready
```

## 5) Test the network

```
gokul@linuxbox: ~/Desktop
$ echo "this is a sample text document" > text.txt
gokul ~/Desktop
$ ipfs add text.txt
added QmFpG6CcDnFwCUpV9f5TzKHq17SVrAPY8rEjLtqIMk7V text.txt
31 B / 31 B [=====] 100.00%
gokul ~/Desktop
$

gokul@debian:~$ ipfs cat QmFpG6CcDnFwCUpV9f5TzKHq17SVrAPY8rEjLtqIMk7V
this is a sample text document
gokul@debian:~$
```

## 6) Explain different Multiformats with example.

Ans).

1. Multihash: While we have various cryptographic hashes, the user can decide which one to use and can easily upgrade when an existing one weakens. It is also easy to add new hash functions, just an additional row in the supported hashes table.

The format used is(type-length-value).

Type is the unique identifier of the hash function used. Example sha1, sha2-256

Length is the fixed length output of hash in bytes

Value is the output of the hash function.

Example:

122041dd7b6443542e75701aa98a0c235951a28a0d851b11564d20022ab11d2589a8

12 represents sha2-256

20 is hex converting to 32 in binary, meaning that the output is 32 bytes

The rest is the digest



2. Multiaddr: When we use just ip address with port number to connect, we do not know the application protocol being used nor the underlying network protocol. Multiaddr overcomes this by using the (type-length-value repeating) format. The value is repeating and some of the examples are

/ipv4/127.0.0.1

/ipv4/127.0.0.1/udp/4023/quic

3. Multibase: We can use different types of base encoding techniques to convert binary to string and multibase helps in identifying which of it is used. The format is (base-encoding-character, base-encoded-data). The examples of base encoding characters are

Base32 -> b

Base32pad -> c

Base64 -> m

Example: BJV2WY5DJMJQXGZJANFZSAYLXMVZW63LFEEQFY3ZP

4. Multicodec: Method used to identify the encoding mechanism applied on the data. The

format used is (encoding-identifier, encoded-data). Some examples of the encoding mechanisms and their corresponding identifiers are

Dag-pb -> 0x70

Dag-cbor -> 0x71

**7. Visit [this link](#). Select an application and explore the same. Please provide your understanding of the application, what problem this solves and how IPFS is used in this application.**

Application selected:

Cyber

Idea of the application:

A decentralized Google alternative, built with the help of IPFS.

Problem it solves:

Google is the most used search engine and most of us derive the information we need through this single entity. The amount of centralized power this gives to google sparks in deciding what we can and cannot see and know sparks the need of an alternative search engine like cyber and its associated protocols.

How IPFS is used:

Queries made to their [site](#) are parsed and relevant results are agreed upon and displayed in an unbiased way via tendermint consensus. The data is stored in IPFS and fetched and user is given the CID as well for him/her to independent verify the authenticity of the content within IPFS.

