GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN IN PUBLICA COMMODA SEIT 1737

HPS

Seminar Report

Emerging Trends in Cloud Storage

Sonal Lakhotia

MatrNr: 14913835

Supervisor: Dr. Julian Kunkel

Georg-August-Universität Göttingen
Institute of Computer Science

March 30, 2023

Abstract

Data fuels almost all businesses today. Data collection, processing, storage, and management enable organizations to efficiently use them for machine learning, risk analysis, or market trend prediction. Digital data storage is primarily of two types Direct-attached Storage (DAS) and Network-based Storage: Network Attached Storage (NAS) and Storage Area Network (SAN). Digital data stored in logical pools called "clouds" is a cloud data storage model called cloud storage. Cloud storage is a cloud computing model that allows data and file storage on the internet through a cloud computing provider. Most cloud storage infrastructures prevalent are centralized, which implies the centralization of data and traffic. This report discusses cloud storage models and types. It also demonstrates and focuses on decentralized cloud architecture, InterPlanetary File System (IPFS), that builds upon Peer-to-peer (P2P) networking and content addressing.

Contents

List of Tables

List of Figures

List of Listings

List of Abbreviations

DAS Direct-attached Storage

NAS Network Attached Storage

SAN Storage Area Network

IPFS InterPlanetary File System

P2P Peer-to-peer

CD Compact disc

USB Universal Serial Bus

API Application Programming Interface

ERP Enterprise Resource Planning

SMB Server Message Block

NFS Network File System

CID Content IDentifier

IPLD InterPlanetary Linked Data

DAG Directed Acyclic Graph

CAR Content Addressable aRchive

DHT Distributed Hash Table

IP Internet Protocol

HTTP Hypertext Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

API Application programming interface

mDNS Multicast Domain Name System

URL Uniform Resource Locator

1 Introduction

Digital data storage enables data processing and retrieval for further tasks of importance. There are two prevalent data storage types: Direct-attached storage and Network-based storage. Direct-attached storage is directly attached to a specific computer. It includes storage space on a computer's hard drive or solid-state drive. It also comprises removable portable storage mediums such as a Compact disc (CD), Universal Serial Bus (USB) drive, flash drive, or hard drive. DAS is advantageous as it includes high availability, easy accessibility, easy backups, and recovery without network equipment or setup. Scaling DAS is also very easy, as it requires acquiring a device with a higher capacity. Data sharing among groups is difficult or virtually impossible with DAS as it requires the physical exchange of the storage drive [Nut]. Network-based storage is a centralized repository of information. Data or information is accessible on networked computers or other networked devices. Servers in a data center store the data rather than a single specified computer. Network-based storage allows multiple users to access information from remote locations. It allows easy data sharing, collaboration, and disaster recovery protection. Cloud storage is a scalable, cost-effective data storage method. It does not require data to be stored on-premise hard drives or on storage networks. Instead, it is stored in offsite locations operated and hosted by cloud service providers. In the rest of the report, cloud storage, its models, types, usage, advantages, disadvantages, and security aspects are discussed in detail. We also discuss a decentralized cloud storage architecture and the working of IPFS.

1.1 What is Cloud Storage?

Cloud storage is a cloud computing model that enables storing data and files over the internet through a cloud computing provider accessible via a public or private network connection. Cloud storage providers ensure secure storage, management, and maintenance of the storage servers, infrastructure, and network so that data is accessible at a virtually unlimited scale with elastic capacity. Cloud storage eliminates the need to buy and manage own data storage infrastructures. It provides scalability, agility, and durability with data access whenever needed [AWS]. Cloud Storage services are accessible through a web service Application programming interface (API), colocated cloud computing service, or applications that use APIs such as cloud-desktop storage or a cloud storage gateway [Wik].

1.2 Importance of Cloud Storage

Cloud storage ensures data redundancy and availability across multiple servers. Although its response time is slower than local storage, it allows data access from any location and is cost-effective. It is easier to set up and maintain cloud storage than local storage. In the following section, advantages of cloud storage are enumerated.

1. **Total cost of ownership**: Cloud Storage empowers organizations to adopt an operational expenditure model rather than capital expenditure and allows them to adjust budgets and resources efficiently.

2. **Elasticity**: Depending on the organization's needs, cloud storage can be scaled up or down. It is elastic and scalable.

3. **Flexibility**: Cloud storage offers flexibility in laying out the architecture of an organization's IT infrastructure. It is flexible in data storage, access, deployment, and budget resources.

4. **Security**: Cloud service providers ensure physical security at data centers and offer robust and cutting-edge security at the application and software levels. They also provide zero trust architecture [Clob], encryption [Cloc], and identity and access management [Clof].

5. **Sustainability**: On-premises data centers encounter the highest costs as overhead energy consumption. Some cloud providers use sustainable energy generated by renewable resources [Clod].

6. **Redundancy**: Redundancy refers to replicating data on multiple servers in different locations. It is an inherent trait in public clouds and allows organizations to recover from disasters while maintaining business continuity.

2 Cloud Storage Architecture

Cloud storage has a highly virtualized infrastructure. Cloud storage services offer off-premises services, such as Amazon S3, and on-premises deployed services, such as ViON Capacity Services [Ser]. The three major cloud storage types are object storage, file storage, and block storage, as seen in Figure Figure 1. All of them have their use cases and advantages.

2.1 How does Cloud Storage Work?

Cloud Storage uses remote servers to store business data, images, and videos. Users upload their data to the servers via an internet connection, and it is saved on a virtual machine in a physical server. The data is spread to multiple virtual machines in data centers located across the world to maintain availability and provide redundancy. More virtual machines are added for an increase in storage. Data access from cloud storage is possible through an internet connection and software such as a web portal or a mobile app via API [Cloe].

Cloud Storage is available in four models:

1. **Public**: It is a cloud storage model where data centers managed by service providers store the organization's data accessible to other organizations all well. The data is spread across multiple regions and is offered on a subscription or pay-as-you-go basis. Public cloud storage is elastic. It implies that data can be scaled as per the organization's needs. Data is made available from devices such as a smartphone or web portals.

2. **Private**: It is a cloud storage model where an organization uses its own servers and data centers to store data on its own network. As an alternative, few organizations deal with cloud service providers to provide servers and private connections that are only dedicated to them and not shared by any other organization. Private clouds are used by organizations that need explicit control over their data and have strict security and compliance requirements.

3. **Hybrid**: Hybrid cloud storage includes private and public cloud storage models. It allows the organization to decide upon data storage in a public or private cloud. Sensitive data that must meet complaint requirements use the private cloud, while less secretive data find storage in a public cloud. Hybrid cloud models have an orchestration to integrate between two clouds. It offers the flexibility to scale up with the public cloud if needed.

4. **Multi-cloud**: When an organization sets up more than one cloud model from multiple service providers (public or private), it forms a multi-cloud storage model. A multi-cloud model offers flexibility and redundancy. It finds use if one cloud vendor offers some proprietary apps, an organization needs data storage in a specific country, different teams need training on multiple clouds, or the organization has to serve requirements not included in the service provider's Service Level Agreements.

2.2 Cloud Storage Types

This section discusses three primary types of cloud storage: object, file and block storage.

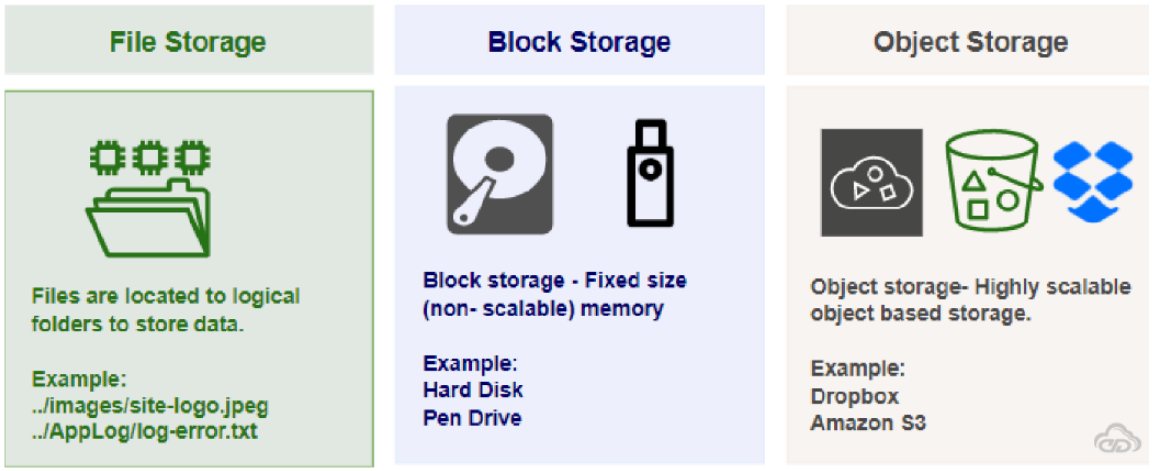

Figure 1: Cloud Storage Types [Stoa]

1. **Object Storage**: Object storage is data storage for large stores of unstructured data, such as photos, videos, audio files, machine learning (ML), sensor data, and other types of web content, in a scalable, efficient, and affordable way. Object storage stores data in its original form and customizes metadata for analysis and accessibility. Objects are stored in secure buckets that offer unlimited scalability instead of files or folders. Hence, it is affordable and economical to save large data volumes. Applications developed in the cloud benefit from the vast scalability and metadata features of object storage. Object storage solutions are perfect for creating modern applications from scratch that require flexibility and import data stores for analytics, archiving, and backup. Object storage use cases are demonstrated in Figure 2

   Object Storage Examples:
   Amazon S3, Oracle Cloud Storage, and Microsoft Azure Storage,object storage software like OpenStack Swift, object storage systems like EMC Atmos, EMC ECS, and Hitachi Content Platform, and distributed storage research projects like OceanStore and VISION Cloud.

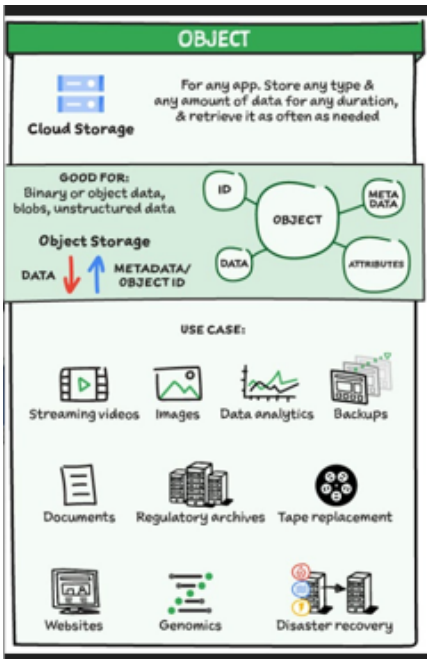

Figure 2: Object Storage [Cloa]

2. **Block Storage**: Databases and Enterprise Resource Planning (ERP) systems need dedicated and low-latency storage for each host, such as DAS or SAN. A cloud storage service that stores data as blocks, and has a unique identifier for each block's rapid storage and retrieval, is most appropriate for such cases. Block storage use cases are demonstrated in Figure 3

   Block Storage Examples:
   Amazon Elastic Block Store (EBS) is used for other enterprise applications like databases and often requires dedicated, low-latency storage for each host.

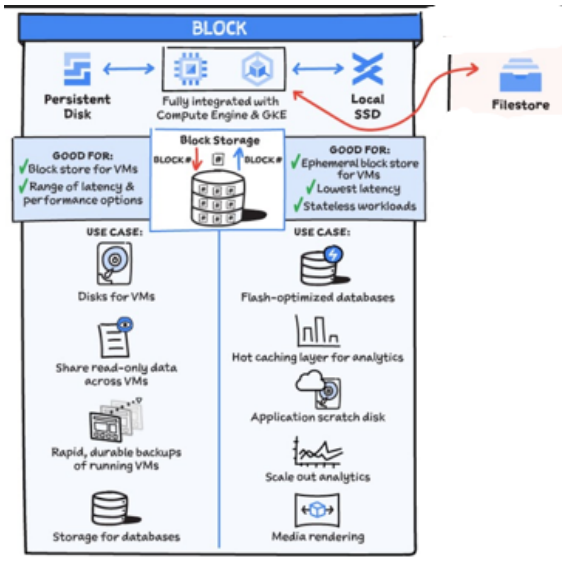

Figure 3: Block Storage [Cloa]

3. **File Storage**: File storage is used by applications and stores data in hierarchical folders and file formats. It is commonly referred to as a NAS server with common

protocols of Server Message Block (SMB) used in Windows instances and Network File System (NFS) found in Linux. File storage use cases are demonstrated in Figure 4

File Storage Examples:
Amazon Elastic File System (EFS) and Qumulo Core, are used for applications that need access to shared files and require a file system.
This storage is often supported with a NAS server, used for large content repositories, development environments, media stores, or user home directories.

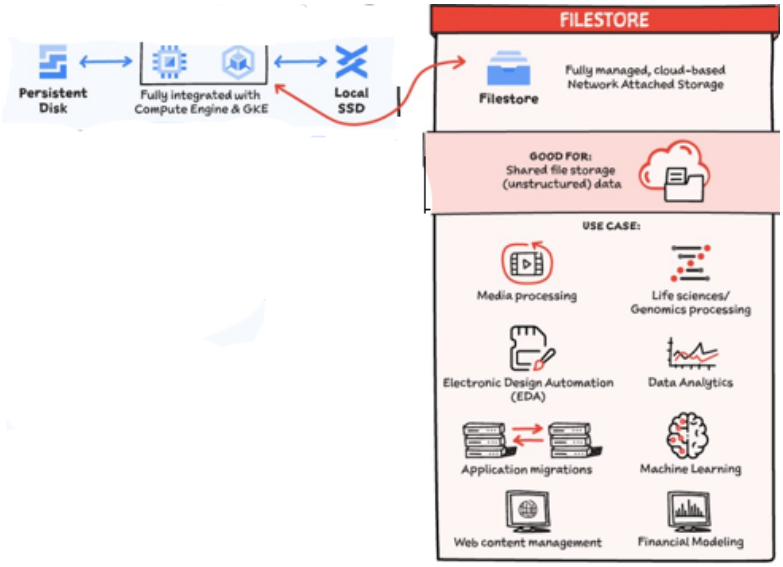

Figure 4: File Storage [Cloa]

3 Cloud Storage Limitations

Although cloud Storage is advantageous in aspects of data scalability accessibility, cost-effectiveness, and many other factors, as we discussed in previous sections, in this section, we discuss a few limitations of cloud storage. In order to combat these limitations, we opt for decentralized storage as discussed in Section 3.1

1. **Compliance**: Finance and healthcare industries have stringent data storage protocols that require public cloud providers to maintain compliance with applicable rules and regulations.

2. **Latency**: Network traffic congestion or slow internet connection delays the traffic to and from the cloud.

3. **Control**: Storing data in public clouds offers control over access and management to the cloud service providers. Consumers entrust the cloud service provider to make data available and maintain its systems and security.

4. **Outages**: Although continuous data availability is ensured by public cloud providers, sometimes outages may make stored data unavailable.

5. **Single Point of failure**: Cloud storage and management have a centralized database. If a database is hacked or crashed, it is possible to lose all data.

3.1 Why use Decentralized Storage?

In a decentralized storage network, several servers hold consumer data. Data security depends on each server and caters to increased reliability and redundancy in case of failures. Unlike centralized cloud storage, there is no single point of failure. In this section, we discuss why decentralized storage is better than centralized cloud storage [Bis].

1. **Increased Security**: Multiple servers store data. Therefore, security depends on each server, not a single server. It prevents files and data from hackers or data loss.
2. **Increased Reliability and Redundancy**: Different servers store each file which minimizes the failure risks and increases redundancy. Hence, no file/data loss occurs due to a crash or a technical error.
3. **Cost**: Minimized data loss reduces cost of data procurement and management which benefits organizations as well as individuals.
4. Examples of decentralized cloud storage: Storj [Stob], IPFS [Vek].

4 What is IPFS?

In this section, we discuss IPFS, the issues it addresses, and it's working. IPFS is a modular set of protocols for transferring and organizing data using peer-to-peer networking and content-addressing principles. IPFS is open-source and has many implementations. IPFS caters to multiple use cases. Its primary use case is to publish files, directories, and websites in a decentralized manner as described in Listing 1

```
ipfs cat /ipfs/QmQPeNsJPyVWPFDVHb77w8G42Fvo15z4bG2X8D2GhfbSXc/about

                IPFS -- Inter-Planetary File system

IPFS is a global, versioned, peer-to-peer filesystem. It combines
good ideas from Git, BitTorrent, Kademlia, SFS, and the Web.
It is like a single bit-torrent swarm, exchanging git objects.
IPFS provides an interface as simple as the HTTP web,
but with permanence built-in. You can also mount the world at /ipfs.
```

Listing 1: About IPFS

IPFS can refer to many concepts:

1. IPFS protocol specification implementation, for example, Kubo.
2. A network composed of IPFS nodes that is decentralized, participatory, and open.

3. Organizing and transferring content-addressed data through a modular set of protocols and standards.

4. IPFS is a protocol, not a provider. Many storage providers have been with IPFS support.

5. IPFS is not a cloud service provider. It can be deployed on cloud infrastructure and complements it.

4.1 Problems Addressed by IPFS

IPFS aims to address problems related to existing web and current data representation/routing/transfer protocols like HTTP [IPF] including:

1. **Verifiability**: IPFS uses cryptographic hashes to verify the integrity and authenticity of files. It becomes challenging for malicious actors to meddle with or delete data.

2. **Resilience**: IPFS does not have a single point of failure. If one or multiple nodes in the IPFS network fail to function, it does not affect the working of the entire network. Allowing IPFS nodes to fetch data from at least one node in the network, irrespective of its location, that has data.

3. **Centralization**: IPFS is more resilient compared to traditional data systems. It is an open, distributed, and participatory network that reduces data silos from centralized servers. IPFS is a community-maintained project. No single person or entity owns or manages data.

4. **Performance**: IPFS allows data replication and retrieval from multiple locations. It provides faster access as users can access data from any location using content addressing instead of location-based addressing. Data access is dependent on its contents. A node fetches data from any other node in the network that has data. Therefore, performance issues like latency do not occur.

5. **Link rot**: IPFS allows data to be addressed by its contents rather than its location. Therefore, link rot problems and dependence on the availability of servers are not an issue.

6. **Data sovereignty**: IPFS enables users to store and access data on a decentralized network of nodes rather than centralized third-party servers. IPFS offers data sovereignty as the need for mediators to manage, own, and control data is eliminated.

7. **Off-chain storage**: IPFS creates a link between the blockchain state and the content addressed published to IPFS, enabling verifiable off-chain storage. A smart contract holds the Content IDentifier (CID).

8. **Local-first software**: IPFS allows data storage, verification, and processing locally. Data synchronization and sharing with other IPFS occurs when the network connection is available. It benefits local-first software and provides a performant, decentralized, peer-to-peer addressing, routing, and transfer protocol with data storage and processing on individual devices as a priority.

9. **Vendor lock-in**: All IPFS users have data and infrastructure sovereignty, thereby preventing vendor lock-in. IPFS is open-source, modular, and community-maintained. Users customize IPFS for their needs, preferred technology, and values. It is not obligated to use a particular subsystem.

4.2 How IPFS works

In this section, we discuss IPFS subsystem composition and its key responsibilities. There are three key roles of IPFS subsystems.

1. **Representing and Addressing Data**
2. **Routing Data**
3. **Transferring Data**

We discuss each of these in detail to understand the underlying subsystems.

4.2.1 How IPFS represents and addresses data

Data is represented as content-addressed blocks and the following subsystems operate on those blocks:

1. **Content Identifier**: In IPFS, data is chunked into blocks and assigned a unique identifier called the content identifier. The hash of the data is combined with its codec, a codec is generated using Multiformats. Using CIDs in IPFS allows data to be fetched based on its content rather than location. The CID of data requested and received can be compared to verify the data requested is the same as the data obtained.
2. **InterPlanetary Linked Data**: IPFS uses InterPlanetary Linked Data (IPLD) to illustrate relationships between content-addressed data, such as file directories and additional hierarchical structures, using a Directed Acyclic Graph (DAG) called a Merkle DAG.
3. **Content Addressable aRchive files**: IPFS utilizes Content Addressable aRchive (CAR) files to reserve and transmit a serialized archive of IPLD content-addressed data.

4.2.2 Content Routing in IPFS

Content routing refers to how IPFS resolves to locate a given CID and peers providing the CID on the network. A node cannot find data with CID alone. It requires knowledge about the IP addresses and ports of the peers on the network. IPFS uses the following subsystems to route content.

1. **Kademlia Distributed Hash Table**: IPFS employs Kademlia,Distributed Hash Table (DHT) created for decentralized peer-to-peer networks. Kademlia enables to locate peers in the IPFS network that hold the data being sought. The Kademlia DHT is a large table dispersed across multiple nodes that stores details concerning peers (Internet Protocol (IP)s) and the data (CIDs) held. Kademlia delivers a favorably efficient, self-organizing technique that resists node churn. Kademlia uses libp2p to establish connectivity.

2. **Bitswap (for content routing)**: IPFS nodes employ Bitswap, a message-based, peer-to-peer network protocol for data transfer and routing. With Bitswap, an IPFS node can ask connected peers for the CID without spanning the Kademlia DHT. Bitswap utilizes libp2p to establish connectivity.

3. **Delegated routing over Hypertext Transfer Protocol (HTTP)**: Delegated content routing is a procedure for IPFS implementations to utilize for off-loading content routing to another process/server employing an HTTP API. Delegated routing over Hypertext Transfer Protocol Secure (HTTPS) provides IPFS nodes with an authoritative interface that allows flexibility in terms of how content routing works.

4. **mDNS**: IPFS uses (mDNS) quickly and efficiently discover peers in local networks. It is a type of DNS protocol that resolves human-readable internet domain names to IP names without the use of a name server.

4.2.3 How IPFS Transfers Data

Data transfer and delivery is handled by the following subsystems:

1. **Bitswap (for data transfer)**: Bitswap enables peers connected to the IPFS node can transfer requested blocks directly to that node without traversing the DHT.

2. **IPFS HTTP Gateways**: HTTP Gateways authorize applications that do not advocate or enforce all IPFS subsystems to fetch data from the IPFS network using an HTTP interface.

3. **Sneakernet**: If data transfer over a network connection is not an alternative, IPFS endorses the use of sneakernet to transmit content-addressed data between IPFS nodes.

5 IPFS Examples

In this section, we demonstrate decentralized web with IPFS and secure file transfer in IPFS.

5.1 Adding a Simple Webpage to IPFS

1. IPFS is installed and initialized.

2. Start the IPFS daemon as shown in Listing 2

3. A new webpage is created in user's home directory

4. The webpage directory is added to IPFS as seen in Listing 3

5. The webpage can be viewed locally as well as on a public gateway. The https://ipfs.io/ domain refers to a public IPFS gateway run by Protocol Labs (the creator of IPFS). A public IPFS gateway provides a HTTP-accessible interface to the Interplanetary File System [Lee].

```
ipfs daemon

Swarm announcing /ip6/::1/tcp/4001
Swarm announcing /ip6/::1/udp/4001/quic
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
```

Listing 2: Starting IPFS daemon

```
ipfs add -r .\simple-webpage\

added QmWsQcVRPygZJGdobZnoL4JmZRTvMuL7LjSJw5dhx4rAi3 simple-webpage/cloud.png
added QmZKjFiXwUbF5ym1TyXUJQTm3gPKBzm8ofECTL9fvTD6EK simple-webpage/index.html
added QmaJ8LtAAsMXWmyjkDVSCZBWLumXwuux4emtxKTjBzDMfN simple-webpage
 279.93 KiB / 279.93 KiB [============================
 ============================================================] 100.00%
```

Listing 3: Adding files to IPFS

6. The webpage can be accessed using the following Uniform Resource Locator (URL):

   `https://ipfs.io/ipfs/QmaJ8LtAAsMXWmyjkDVSCZBWLumXwuux4emtxKTjBzDMfN/`

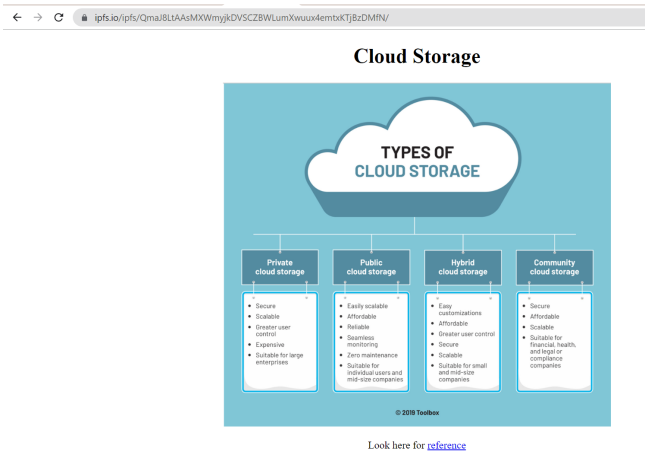

Figure 5: A simple webpage in IPFS

7. Figure 5 shows a simple webpage hosted in IPFS. It is content-addressed and can be located from anywhere.

5.2 Securing Adding Files to IPFS

All files in IPFS are public. If we want files to be accessed only by a few people or a particular set of people, we could encrypt and upload them to IPFS. Figure 6 shows the normal workflow in IPFS. A file is requested by the receiver, and it is easy to access the file using hash sent by the sender. Figure 7 shows secure file sharing in IPFS using asymmetric encryption.

Figure 6: File Transfer in IPFS [Hea]

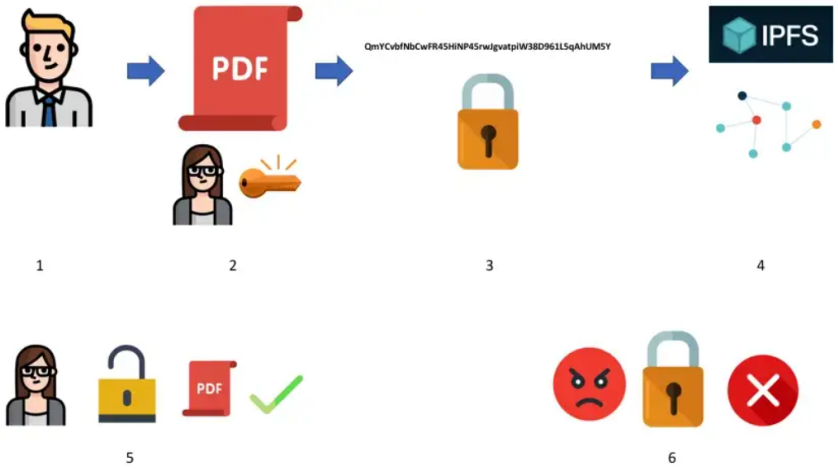

Figure 7: Secure File Transfer in IPFS [Hea]

In order to securely transfer files with IPFS the following steps are carried out:

1. Both users set up GPG and the requester sends the GPG key to the sender so that it can be added to the sender's keyring. Files encrypted by this key can only be decrypted by the requester it is intended for.

2. Successful key import can be checked as shown in Listing 4

```
C:\Users\jssso\OneDrive\Desktop\IPFS-playground> gpg --import pubkey.asc
gpg: key 22DD53420A34965A: public key "Akarsh<akarshanurag1996@gmail.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1


C:\Users\jssso\OneDrive\Desktop\IPFS-playground> gpg --list-keys
C:\Users\jssso\AppData\Roaming\gnupg\pubring.kbx
pub ed25519 2023-01-29 [SC] [expires: 2025-01-28]
    17BC1BEBDF5CD26C86F78D3F8C62FFCCB43EF841
uid     [ultimate] Sonal <jsssonallakhotia716@gmail.com>
sub cv25519 2023-01-29 [E] [expires: 2025-01-28]
pub ed25519 2023-01-29 [SC] [expires: 2025-01-28]
    E2EF6926B4532DA2D822B1A622DD53420A34965A
uid     [ unknown] Akarsh <akarshanurag1996@gmail.com>
sub cv25519 2023-01-29 [E] [expires: 2025-01-28]
```

Listing 4: Listing imported keys

3. IPFS is setup and daemon is started as seen in Listing 2
4. The file to be sent is encryped as seen in Listing 5

```
gpg --encrypt --recipient "Akarsh" IPFS.txt
gpg: 2EA8CBA4EA589765: There is no assurance this key belongs to the named user

sub cv25519/2EA8CBA4EA589765 2023-01-29 Akarsh
Primary key fingerprint: E2EF 6926 B453 2DA2 D822 B1A6 22DD 5342 0A34 965A
Subkey fingerprint: B2E1 016C 4EB7 07DA D576 4227 2EA8 CBA4 EA58 9765

It is NOT certain that the key belongs to the person named in the user ID.
If you *really* know what you are doing, you may answer the next question with yes.
Use this key anyway? (y/N) y
```

Listing 5: Encrypting file to be sent

5. The encrypted file is available in the directory, the file is uploaded to IPFS as shown in Listing 6
6. The requester downloads the file as seen in Listing 7 using the same hash as obtained in Listing 6
7. The file is decrypted to be viewed by an authenticated user as seen in Listing 7 line 7.
8. Decrypted file contents can be viewed securely the requester as Using IPFS secure file transfer is possible using asymmetric encryption.

```
ipfs add IPFS.txt.gpg

added QmWxjZpzTGFELGZtod6m9gej h2RDoCtXsT5Uzptmxbp9sf IPFS.txt.gpg
3.79 KiB / 3.79 KiB [=====================================] 100.00%
```

Listing 6: Uploading encrypted file to IPFS

```
ipfs get QmWxjZpzTGFELGZtod6m9gejh2RDoCtXs T5Uzptmxbp9sf

Saving file(s) to QmWxjZpzTGFELGZtod6m9gejh2RDoCtXsT5Uzptmxbp9sf
3.79 KiB / 3.79 KiB [=====================================] 100.00% 0s


gpg --decrypt QmWxjZpzTGFELGZtod6m9gejh2RDoCtXsT5Uzptmxbp9sf > IPFS.txt
gpg: encrypted with cv25519 key, ID 2EA8CBA4EA589765, created 2023-01-29
```

Listing 7: Downloading the encrypted file

6 Future Work

IPFS presents various opportunities for current projects and future developments. Infrastructure to support IPFS and browser support for IPFS would enable a full-fledged use for IPFS and a stable web [Obe]. Currently, it is used as off-chain storage for Ethereum, where the transaction contains an immutable IPFS CID while the data of the respective CID is stored on IPFS [Doa+22]. Other projects include content delivery over persistence/governance to social media and e-commerce, such as Textile.io [Hil]. Project Filecoin endeavors to solve the lack of incentive for data storage in IPFS content storage.

7 Conclusion

The report addresses the importance of data procurement, storage, and management. Cloud storage offers many advantages for data storage and retrieval. It provides secure and affordable storage for massive quantities of business data. We discuss cloud storage models and utilities, file storage, object storage, and block storage, their examples, and their use cases. The report emphasizes the benefits of cloud storage but also illustrates their limitations and disadvantages. It demonstrates the motivation and advantages of decentralized storage. Decentralized cloud storage architecture IPFS is discussed in detail, and its working principles are highlighted. IPFS examples show the usage and implementation for content-based addressing and secure file transfer using asymmetric encryption in IPFS. We conclude that better infrastructure and projects would enable greater adoption of IPFS and pave way for a stable and decentralized web. My learning and understanding about emerging cloud storage techniques enhanced while working on this report.

References

[AWS] AWS. "What is Cloud Storage". In: (). URL: https://aws.amazon.com/what-is/cloud-storage/.

[Bis] Nuno Bispo. "The Benefits of Decentralized Storage vs Cloud Storage". In: (). URL: https://medium.com/geekculture/the-benefits-of-decentralized-storage-vs-cloud-storage-f5f01592ed9d.

[Cloa] Google Cloud. "A map of storage options in Google Cloud". In: (). URL: https://cloud.google.com/blog/topics/developers-practitioners/map-storage-options-google-cloud.

[Clob] Google Cloud. "BeyondCorp Enterprise". In: (). URL: https://cloud.google.com/beyondcorp-enterprise.

[Cloc] Google Cloud. "Cloud Key Management". In: (). URL: https://cloud.google.com/security-key-management.

[Clod] Google Cloud. "Cloud Sustainability". In: (). URL: https://cloud.google.com/sustainability.

[Cloe] Google Cloud. "How does Cloud Storage Work". In: (). URL: https://cloud.google.com/learn/what-is-cloud-storage#section-3.

[Clof] Google Cloud. "Identity and Access Management (IAM)". In: (). URL: https://cloud.google.com/iam.

[Doa+22] Trinh Viet Doan et al. "Towards Decentralised Cloud Storage With IPFS: Opportunities, Challenges, and Future Considerations". In: *IEEE Internet Computing* (2022).

[Hea] Coral Health. "Learn to securely share files on the blockchain with IPFS!" In: (). URL: https://mycoralhealth.medium.com/learn-to-securely-share-files-on-the-blockchain-with-ipfs-219ee47df54c.

[Hil] Andrew Hill. "Adding the next million peers to IPFS". In: (). URL: https://medium.com/textileio/adding-the-next-million-peers-to-ipfs-76d356352d14.

[IPF] IPFS. "What is IPFS?" In: (). URL: https://docs.ipfs.tech/concepts/.

[Lee] Wei-Meng Lee. "Understanding the InterPlanetary File System (IPFS)". In: (). URL: https://blog.cryptostars.is/understanding-the-interplanetary-file-system-ipfs-147e5d209d56.

[Nut] Nutanix. "What is Data Storage: Types of Data Storage". In: (). URL: https://www.nutanix.com/blog.

[Obe] Matt Ober. "The IPFS Cloud". In: (). URL: https://medium.com/pinata/the-ipfs-cloud-352ecaa3ba76.

[Ser] ViON Capacity Services. "ViON". In: (). URL: https://web.archive.org/web/20160322022215/http://www.vion.com/capacity-services/vion-capacity-services.html.

[Stoa] Cloud Storage. "Different types of AWS Cloud Storage – Cloud Storage Quick Guide". In: (). URL: https://cdcloudlogix.com/different-types-of-aws-cloud-storage/.

[Stob] Storj. “Fast, secure cloud storage at a fraction of the cost.” In: (). URL: https://www.storj.io/.

[Vek] Maxim Veksler. “IPFS-Inter-Planetary File system”. In: (). URL: https://medium.com/@mvxlr/ipfs-inter-planetary-file-system-65466e4129c6.

[Wik] Wiki. “Cloud Storage”. In: (). URL: https://en.wikipedia.org/wiki/Cloud_storage.

A Work sharing

If you worked in a group, describe here how you distributed the work and the actual contributions of each peer.

A.1 Hans

...

A.2 Peter

...

B Code samples

This is part of the appendix...