Data Provenance Assurance for Cloud Storage using Blockchain

Abhishekh Patil, Amit Jha, Mohammed Moin Mulla, Narayan D.G.[1], Shivaraj Kengond

School of Computer Science and Engineering, KLE Technological University, Hubli, Karnataka, India
abhishekpatil0853@gmail.com, ajha1054@gmail.com, moin.mulla@kletech.ac.in,
narayan_dg@kletech.ac.in, shivaraj.kengond@kletech.ac.in

***Abstract*: Cloud forensics investigates the crime committed over cloud infrastructures like SLA-violations and storage privacy. Cloud storage forensics is the process of recording the history of the creation and operations performed on a cloud data object and investing it. Secure data provenance in the Cloud is crucial for data accountability, forensics, and privacy. Towards this, we present a Cloud-based data provenance framework using Blockchain, which traces data record operations and generates provenance data. Initially, we design a dropbox like application using AWS S3 storage. The application creates a cloud storage application for the students and faculty of the university, thereby making the storage and sharing of work and resources efficient. Later, we design a data provenance mechanism for confidential files of users using Ethereum blockchain. We also evaluate the proposed system using performance parameters like query and transaction latency by varying the load and number of nodes of the blockchain network.**

Keywords— Cloud Storage; AWS S3; Data Provenances; IPFS; Ethereum.

I. INTRODUCTION

Cloud storage is a technology that uses the Internet and Remote Servers to maintain data and applications. It is one of the core technologies behind many online services for personal applications. This technology can be used on a small scale in universities. It can also be used in colleges to virtualize resources, making use of hardware more efficient. As such, cloud storage applications can become an important source of online education programs, e-learning systems, and mobile learning.

Cloud forensics investigates the crime committed over cloud infrastructures like SLA-violations and storage privacy [1]. Cloud forensics investigates the crime committed over cloud infrastructures like SLA-violations and storage privacy. Cloud storage forensics is a process of data provenance which determines the history of a storage object, starting from its creation [2]. Data provenance assurance in cloud environments helps in the cloud forensics. However, designing assured data provenance models is an important issue for cloud storage. Furthermore, provenance data may contain sensitive information about the original data and the data owners. Thus there is a need to secure cloud data with both integrity and trustworthiness of provenance data [3].

Blockchain technology has gained a lot of importance due to a shared, distributed ledger which records transactions across many network nodes so that any involved record cannot be altered retroactively, without the alteration of all subsequent blocks. Blockchain's decentralized architecture helps in developing an assured data provenance capability in cloud environments. In a cloud environment, there is a need to safeguard personal data while maintaining privacy. Using Blockchain, data operations can be transparently and permanently recorded. Furthermore, maintaining provenance can assist in improving the trust of cloud users about secure information sharing [4] [5].

In this paper, we proposed a data provenance mechanism using Blockchain and IPFS. This mechanism assures the user files stored in cloud storage. We develop an AWS S3 storage application to demonstrate the process. The storage application records the operations of the user like add, delete, edit for user-specified files. These log records are stored in the IPFS network, which gives the Hash. These hashes of provenance data are then stored in Ethereum blockchain networks as a transaction. We also validate the mechanism if an attempt is made to modify a provenance data record. The main contributions of the proposed work are as follows.

- Developed a Cloud Storage web Application for an AWS S3 cloud.
- Designed a mechanism to store provenance data using Ethereum Blockchain and IPFS.
- Carried out the scalability and performance analysis of the proposed mechanism using different scenarios.

The rest of the paper is organized as follows. Section II discussed the related work on data provenance security assurance using blockchain technology. Section III describes the design of cloud storage applications on OpenStack based private Cloud and blockchain-based cloud storage data provenance architecture. Performance

978-1-7281-9785-2/20/$31.00 ©2020 IEEE

evaluation of architecture is presented in Section V. Finally, we conclude in Section VI

II. RELATED WORK

Authors in [1] have proposed an architecture that is known as provchain. This architecture focused on collecting and verifying the provenance of Cloud by anchoring the data into the blockchain transactions. It [2] uses an open-source, public cloud called ownCloud to get the provenance data. It also uses a blockchain API called Tieron data API to connect to the node of Blockchain. Authors in [3] have identified the operations performed by the user as a unit to get provenance data. Authors proposed a system that made the provenance data available widely, auditable, and its contents provable, thereby providing trust between user and Cloud. This paper made use of Ethereum blockchain and OpenStack Swift for storage applications.

Authors in [4] have proposed anti-tampering and privacy preservation for the cloud forensics. This process provides proof of the existence of process records. The authors proposed an architecture that has Merkle root for printing blockchain receipts as it is helpful for data validation. Authors in [5] proposed architecture known as CloProv, which is indeed a model that captures the provenance of any type of entity in the Cloud. They also proposed a blockchain type architecture called SecProv and integrated with the OpenStack Swift based Storage application. The authors also proposed an analysis of the efficiency of the proposed scheme.

Many researchers have carried out work in the domain of data provenance in the Cloud. In [6], authors develop an end to end data tracking tool which provides both file-level and block-level provenance in kernel space. Further, the authors explore the security of provenance data and user privacy. In [7], the authors propose a secure data provenance mechanism in the cloud environment which adopts a two- folder encryption method to enhance privacy, incurring a higher computation cost. In [8], the authors protect provenance data confidentiality and integrity using encryption and digital signature. However, SPROVE does not possess provenance data querying capability. In [9], authors develop a kernel-level logging tool which can provide log tamper-evidence at the expense of user privacy. However, all the work carried above is without using Blockchain.

In [10], the authors use the Tierion platform for uploading data into the Blockchain network using API's. APIs help in integrating web applications to blockchain networks. With public APIs available, Tierion is convenient for integrating applications that demand the need of Blockchain. Tierion is used to store and fetch the data using HTTP. Blockstack Labs propose another data provenance mechanism from Princeton University in [11]. Authors use decentralized PKI service and a Blockchain-based naming and storage system for provenance. The authors in [12] proposed blockchain application in an information-centric network for name-based security of content distribution has also been proposed. In [13], the authors proposed another mechanism called Enigma with guaranteed privacy using Blockchain. In [14], the authors proposed blockchain services using Keyless Signature Infrastructure (KSI) and hash function. The authors also proposed a blockchain standard for digital identity and a protocol for authentication and digital signature [15]. In [16], the authors design a Blockchain-based data provenance mechanism called ProvChain for cloud auditing. The mechanism provides, with user privacy and increased availability for cloud storage data. Authors use open source storage applications to demonstrate the mechanism. However, authors have not used any cloud operating system for evaluation. As storage is one part of cloud infrastructure, the evaluation of storage applications with cloud operating systems is an important issue.

III. PROPOSED WORK

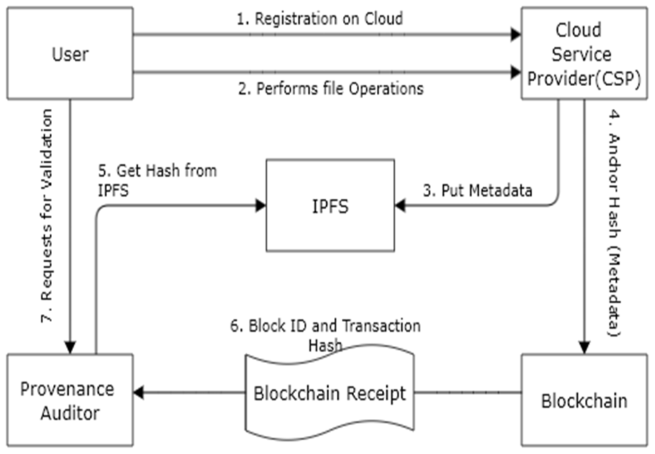
In this section, we discuss the design of storage application and data provenance assurance using Blockchain.

A. Storage Application

We use Amazon Web Services (AWS) SDK to develop storage applications. We used Boto3 model of AWS SDK for Python. Before using Boto3, we set up authentication credentials, and generate a new set of keys, and we accessed boto3 SDK and uploaded the files in the Simple Storage Service (S3). We choose the IAM (Identity access management) service and then got an access key and the access id and configured the access key and the access id from settings in the Django. Next, we created a bucket and then implemented boot3, which has commands like upload, delete, and download; these are the commands provided by the boot3. Finally, we upload all the files which need to be uploaded from boot3 to s3. We also designed a front-end for the application. Using this storage application, we created a provenance data for storing in blockchain networks.

B. Data Provenance Assurance

The below Fig. 1 illustrates the proposed system of data provenance assurance, a user creates an account and logs into the application using his/her private key. The user is authorized to perform mining operations upon successful login. When the user performs read, write, or any operations on any file, it triggers to generate metadata, which is anchored to the Blockchain network and IPFS. The provenance database manages the IPFS. Once the block is mined the Blockchain does Validation, the generated receipt will be returned to the provenance database. The receipt is banded with the provenance entry in the IPFS. The provenance database audits every user's activities but can never identify the valid owner, as it stores the hashed identity of the users.



User
1. Registration on Cloud
2. Performs file Operations
Cloud Service Provider(CSP)
7. Requests for Validation
5. Get Hash from IPFS
IPFS
3. Put Metadata
4. Anchor Hash (Metadata)
6. Block ID and Transaction Hash
Provenance Auditor
Blockchain Receipt
Blockchain

Fig. 1: Data Provenance Assurance Mechanism

The below Fig. 2 shows the system's layered architecture, which integrates two modules, i.e., Cloud and Blockchain, in the Presentation layer. The users interact with the system using the web interface layer. The authentication layer acts as a business layer. The service layer provides options to perform operations to users. Database layer consists of IPFS and Blockchain. IPFS is used to store metadata as a database for the credentials and Blockchain as securely storing the provenance data.

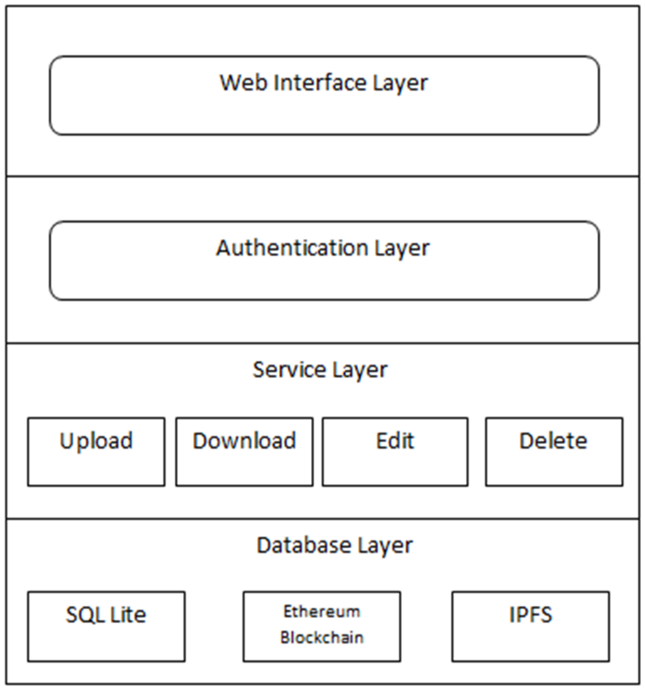


Fig. 2: Architecture of System

B. Data Storage Provenance Algorithms

We propose three algorithms to implement Data storage provenance such as Retrieval of Metadata, Storing in the Blockchain and ipfs, and Validation if any discrepancy.

Algorithm 1: Retrieval of Metadata

```
Input: File operations done by the cloud user.
Output: Metadata is generated.
1.  For each cloud user do
2.      For each file operations done do
3.          Generate_metadata (file_ operation, metadata)
4.      End for
5.  End for
6.  Function Generate_metadata (file_
    operation,metadata)
7.          User_id←user_id
8.          Action←action_performed
9.          Timestamp←timestamp
10. End
```

Algorithm 1 generates metadata manually. Generally, CSP has a logger, which maintains logs of all the operations performed by the user and stores the metadata of the operations. Amazon S3 SDK doesn't provide any logger, so the metadata has been generated manually by the server. When the user performs any operations on any files, the log is added to IPFS and subsequently to the Blockchain. Input is the operation performed by the user (create, edit, delete) and output is metadata log is generated.

Algorithm 2 : Storing in the Blockchain and ipfs

```
Input: Metadata.
Output: Update database with operation details.
1.  If user is authorized then
2.    For each file operations done do
3.      Metadata=Get_metadata(file_operation,metadata)
4.      Hash_from_ipfs=add_to_ipfs(Metadata)
5.      Block_id=Store_it_in_blockchain(Hash(Provenan
        ce_entry)
6.          Update_mapping_file(Hash(user_id), Hash_fr
            Om_ipfs,Block_id,time_stamp)
7.    End for
8.  End if
9.  End
```

The algorithm 2 processes the collected provenance data, i.e., metadata, will be further added to IPFS, and its hashed value will be added to the Blockchain. For each file operation, metadata will be generated, which will be added to IPFS and Blockchain.

Algorithm 3: Validation if any discrepancy.

```
Input: User requests for validation (mapping_file).
Output: True/False.
1.  Bool validate_file(Report)
2.      For all entry M in mapping_file do
3.          If(found(Block_id)) then
```

```
4.              X=get_hash_from_Blockchain(Block_id)
5.              Y=get_value_from_IPFS(hash_from_ipfs)
6.                  If(X!=hash(Y)) then
7.                      Return False;
8.                  End if
9.           End if
10.      End for
11.   Return True;
12.  End
```

The algorithm 3 is of Provenance Validator. It gets the report from the server to check for Validation of the integrity and discrepancy of the files. For all entries in mapping files, Provenance Validator gets block id and checks the hashed metadata from the Blockchain with the metadata from the IPFS. If the values corresponding to all the rows match, if the returned value is true, then the data has not tampered; if the returned value is false, then the data has tampered. By using this technique, we can easily find the tampering of the data.

IV. RESULTS AND DISCUSSION

This section presents the performance evaluation of our proposed data storage provenance using Blockchain. The results are analyzed for IPFS and Blockchain. The results were analyzed concerning latency by varying no of nodes, load, and difficulty level.

A. *Experimental Setup*

For evaluating the performance of Ethereum transactions, we have used the BaaS service deployed in the OpenStack cloud of the campus. For the experimental purpose, three VMs with the configurations of 2GB RAM, 2 VCPUs, 20GB HDD is used. We used AWS S3 storage to get the logs. AWS S3 is a storage component, where we can store the files or any kind of data. In this setup, AWS S3 is used to store the logs. We have used Web 3 API for connecting Ethereum nodes. It is used to send several bunches of these logs (transactions) to the Blockchain using a smart contract and capture the processing time for queries to get executed and return a hash for the block. The details system configurations are shown in Table 1.

Table 1: System Configuration

Component	Tool/Utility	Version
Operating System	Ubuntu	16.04 LTS
Blockchain Network	Ethereum	ETH
Blockchain Client	Geth	1.7.2
Interface		
Smart Contract	Solidity	4.17
Distributed File System	IPFS	6.0

B. *Performance Analysis of Ethereum Blockchain*

In this section, we evaluate the impact of the blockchain network in providing data provenance. This is very important from a scalability and performance point of view. We discuss the results using three scenarios as follows.

- *Impact of Load*

The primary function of Blockchain is performing transactions. A transaction could be either reading or writing the data into the chain. In the below table 2, the transaction represents writing data into the Blockchain. Proof of Work consensus algorithm with a difficulty level of 0x00 kept as constant. We have experimented with 5, 10, 15, 20 transactions. We observed that the increase in load or transaction latency increases linearly. Therefore work done is proportional to the time taken.

Table 2: Impact of Load

No of Transactions	5	10	15	20
Read Latency (Secs)	0.9	1.39	1.99	2.70
Add Latency (Secs)	45.9	170.43	304.89	417.9

- *Impact of Network Size*

One of the reasons for the need for scaling Blockchain is the growing number of nodes over time and will continue to grow. This leads to consequences like an increase in network transaction fees; thus, latency also increases, but read latency almost remains constant. The below table 3 explains the behavior of latency over the number of nodes. We have increased the number of nodes from 1 to 3. Generally, blockchain networks will have a large number of nodes which maintain distributed ledger. Thus, it impacts the performance of the proposed data provenance mechanism.

Table 3: Impact of Network Size

No of Nodes	1	2	3
Read Latency (Secs)	0.027	0.03	0.03
Add Latency (Secs)	3.48	23.97	45.9

- *Impact of Difficulty level*

A difficulty level is a number that denotes the number of zeroes that need to be appended before the Hash. The significance of difficulty is to make miners work more complex by solving a more complex puzzle. Therefore miners have to use their CPU computational power to get Hash with a specified number of zeroes. As shown in Fig. 3, as difficulty level increased, the transaction latency became more. Two node setup has been used for this scenario.

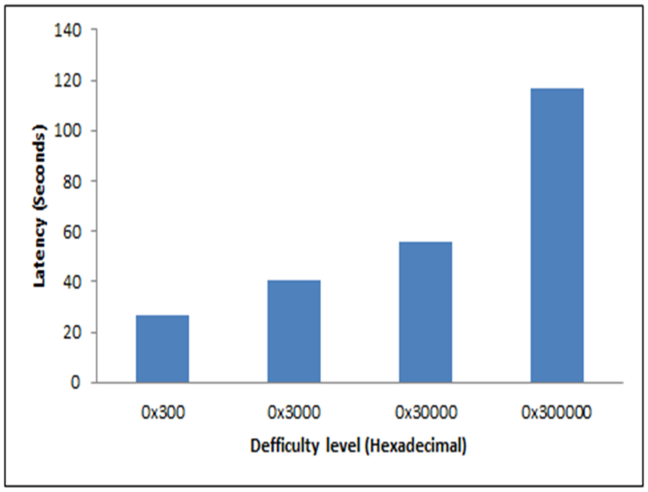


Fig 3: Transaction latency vs Difficulty level

C. IPFS Results Analysis

IPFS is the distributed storage used in our system. We store the log records in the IPFS. The performance of this operation is important in the design process. Thus, we analyze the results of IPFS read latency with varying file size. Fig 4. Concludes that the large read requests, the read latency increases with request sizes, since the local node needs to read more blocks.

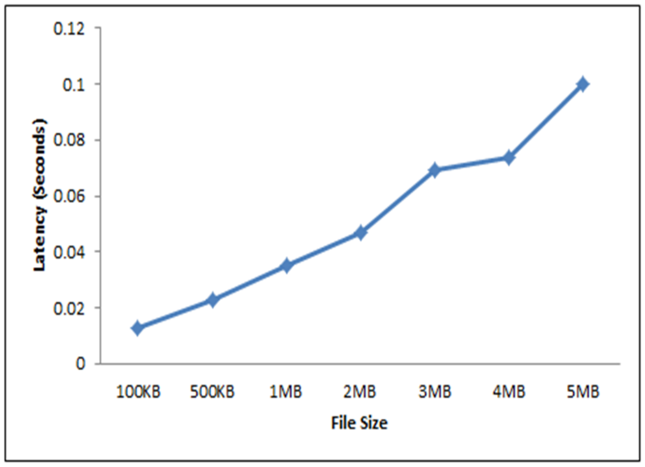


Fig. 4: Read Latency of IPFS

We analyze the results of IPFS add latency with varying file size. Fig 5. Concludes that for small requests, IPFS can quickly finish the write operation in the local node. For the large requests (>1MB) the Writing latency increases with the request size, since IPFS divides a file into multiple blocks. This incurs overhead with hard disk writes while storing blocks in cloud storage.

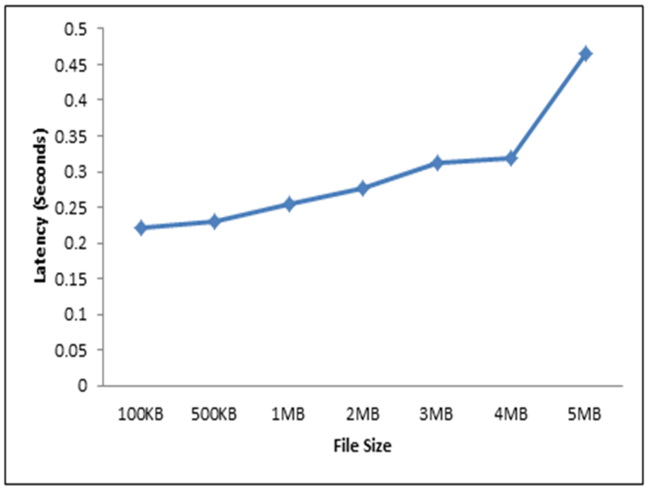


Fig. 5: Latency for Adding to IPFS

V. CONCLUSION AND FUTURE WORK

In this paper, we presented the design and implementation of a data provenance assurance mechanism using Blockchain and IPFS. Initially, we developed a cloud storage application using AWS S3 API. This application logs the provenance data and stores Hash of it in an Ethereum based distributed ledger. IPFS gives the Hash of the record for the log record. The user specifies the files for data provenance. This decentralized mechanism helps in providing trust among the user and CSP. We also carried out the scalability and performance analysis of the proposed system using different scenarios.

As future work, we plan to add security to the mapping file and provenance service to all the other files uploaded by the user and perform scalability and performance analysis with a larger blockchain network.

REFERENCES

[1] Aditya C., Akash M., Akash P., Amitkumar M., Nagarathna K., Suraj D., Narayan D.G., Meena S.M., Claims-Based VM Authorization on OpenStack Private Cloud using Blockchain, Procedia Computer Science, Volume 171, 2020, Pages 2205-2214, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2020.04.238.

[2] Y. L. Simmhan, B. Plale, and D. Gannon, "A survey of data provenance in e-science," *ACM Sigmod Record*, vol. 34, no. 3, pp. 31–36, 2005.

[3] B. Lee, A. Awad, and M. Awad, "Towards secure provenance in the cloud: A survey," in *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*. IEEE, 2015, pp. 577–582.

[4] D. Tosh, S. Sengupta, C. A. Kamhoua, and K. A. Kwiat, "Establishing evolutionary game models for cyber security information exchange (cybex)," *Journal of Computer and System Sciences*, 2016.

[5] C. Kamhoua, A. Martin, D. K. Tosh, K. Kwiat, C. Heitzen- rater, and S. Sengupta, "Cyber-threats information sharing in cloud computing: A game theoretic approach," in *IEEE 2nd International Conference on Cyber Security and Cloud Computing (CSCloud)*, 2015, pp. 382–389.

[6] C. H. Suen, R. K. Ko, Y. S. Tan, P. Jagadpramana, and B. S. Lee, "S2logger: End-to-end data tracking mechanism for cloud data provenance," in *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*. IEEE, 2013, pp. 594–602.

[7] M. R. Asghar, M. Ion, G. Russello, and B. Crispo, "Securing data provenance in the cloud," in *Open Problems in Network Security*. Springer, 2012, pp. 145–160.

[8] R. Hasan, R. Sion, and M. Winslett, "Sprov 2.0: A highly-configurable platform-independent library for secure prove- nance," *ACM, CCS, Chicago, IL, USA*, 2009.

[9] R. K. Ko and M. A. Will, "Progger: An efficient, tamper- evident kernel-space logger for cloud data provenance track- ing," in *2014 IEEE 7th International Conference on Cloud Computing*. IEEE, 2014, pp. 881–889.

[10] "Tierion api," https://tierion.com/app/api. Last Accessed [20th July, 2020]

[11] M. Ali, J. Nelson, R. Shea, and M. J. Freedman, "Blockstack: A global naming and storage system secured by blockchains," in *2016 USENIX Annual Technical Conference (USENIX ATC 16)*, 2016.

[12] N. Fotiou and G. C. Polyzos, "Decentralized name-based security for content distribution using blockchains," in *Com- puter Communications Workshops (INFOCOM WKSHPS), 2016 IEEE Conference on*. IEEE, 2016, pp. 415–420.

[13] G. Zyskind, O. Nathan, and A. Pentland, "Enigma: Decen- tralized computation platform with guaranteed privacy," *arXiv preprint arXiv:1506.03471*, 2015.

[14] A. Buldas, A. Kroonmaa, and R. Laanoja, "Keyless signatures infrastructure: How to build global distributed hash-trees," in *Nordic Conference on Secure IT Systems*. Springer, 2013, pp. 313–320.

[15] A. Buldas, R. Laanoja, and A. Truu, "Efficient implementation of keyless signatures with hash sequence authentication." *IACR Cryptology ePrint Archive*, vol. 2014, p. 689, 2014.

[16] D. Tosh, S. Shetty, X. Liang, C. Kamhoua and L. L. Njilla, "Data Provenance in the Cloud: A Blockchain-Based Approach," in *IEEE Consumer Electronics Magazine*, vol. 8, no. 4, pp. 38-44,July,2019.