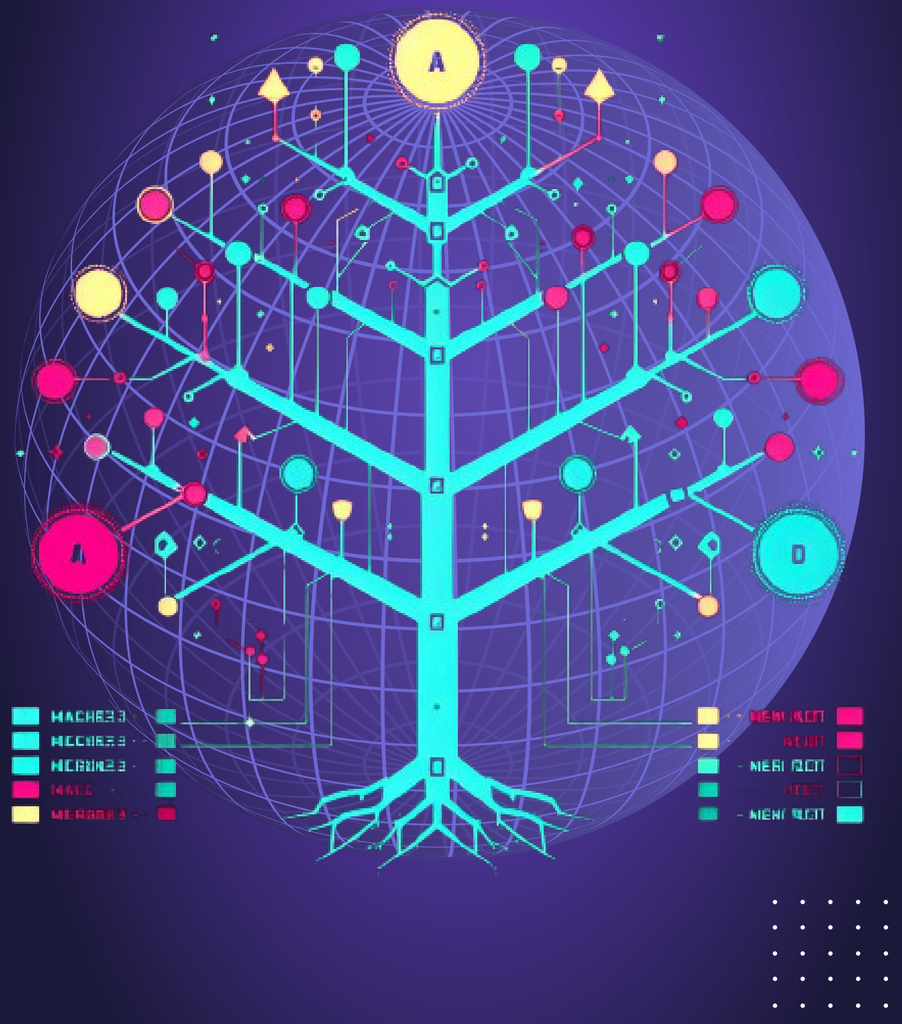
001

MERKLE TREE

DATA STRUCTURE



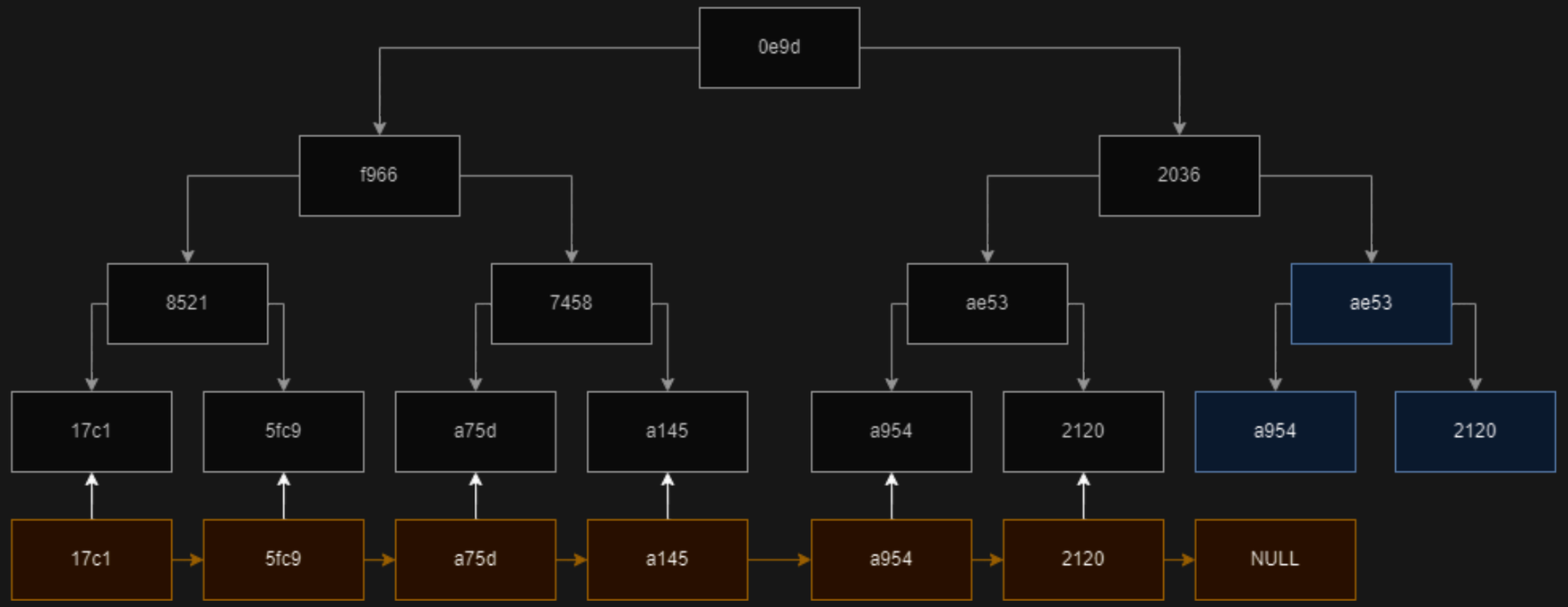
WELCOME TO CLASS!

Today's Agenda

- Introduction
- Merkle Proof
- Use cases
- Second Preimage Attack
- Complexity
- Implementation
- Conclusion

INTRODUCTION

Merkle Tree or Hash Tree



MERKLE PROOF

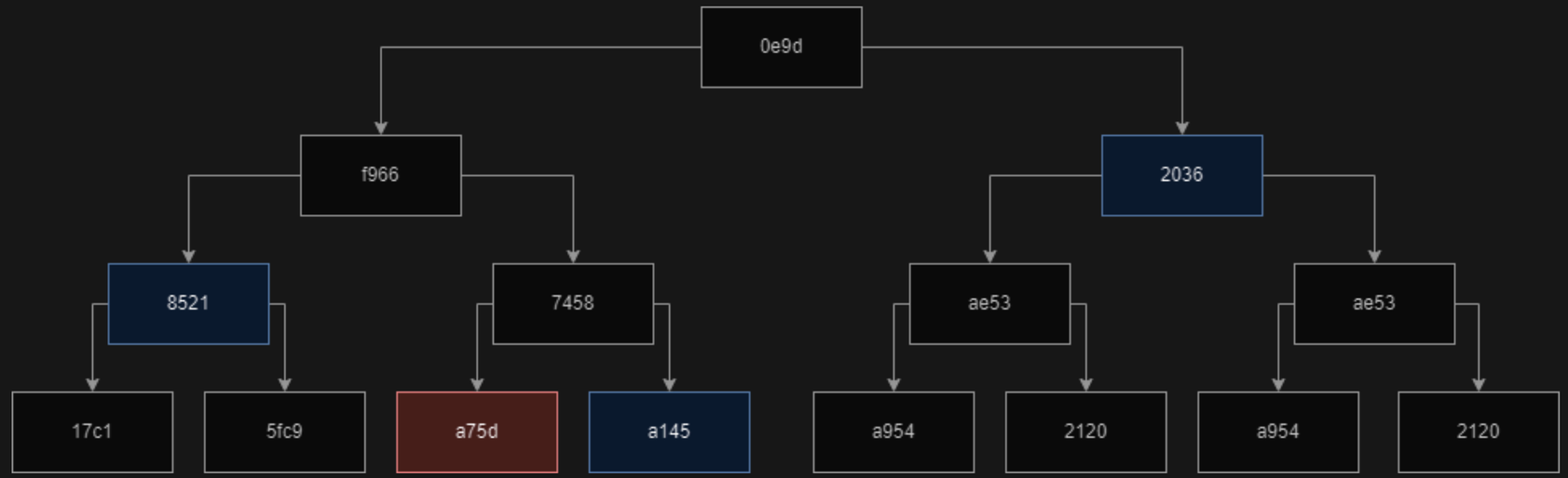
Merkle Path and minimum proofs

For merkle root [0e9d] and leaf [a75d]

Merkle Path: [a75d, a145, 8521, 2036]

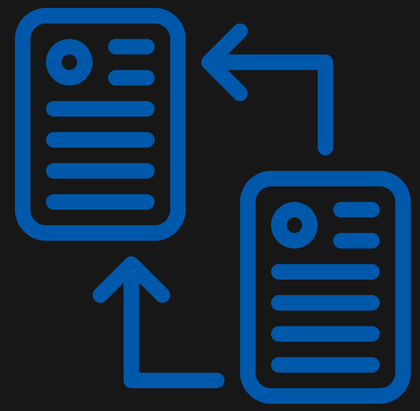
Pattern on validation

We only need *log(n)* elements of the tree to check a hash.



USE CASES

How does this work?



HASH-BASED CRYPTOGRAPHY

Digital signatures schemes based on Merkle signature

GIT AND MERCURIAL

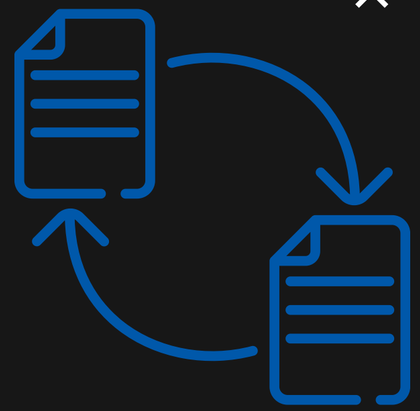
Distributed revision control systems

BITCOIN AND ETHEREUM

Peer-to-peer network

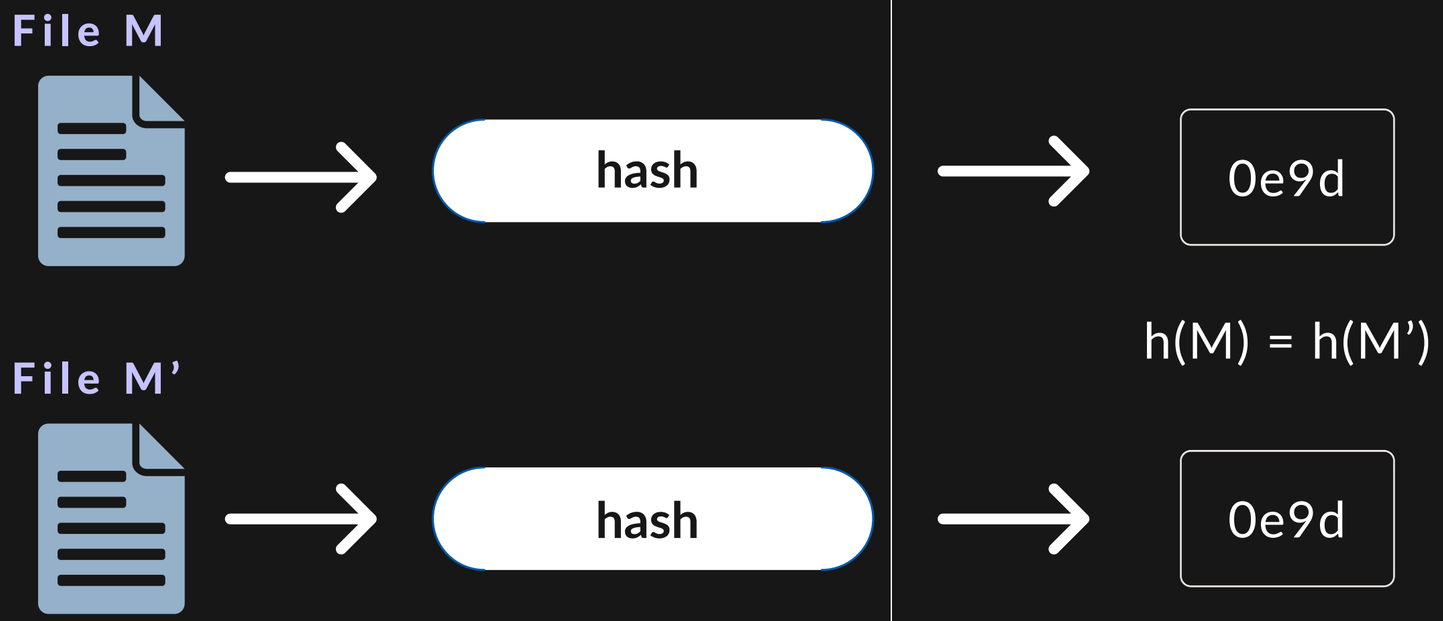
NOSQL SYSTEMS

Find inconsistencies in replicas



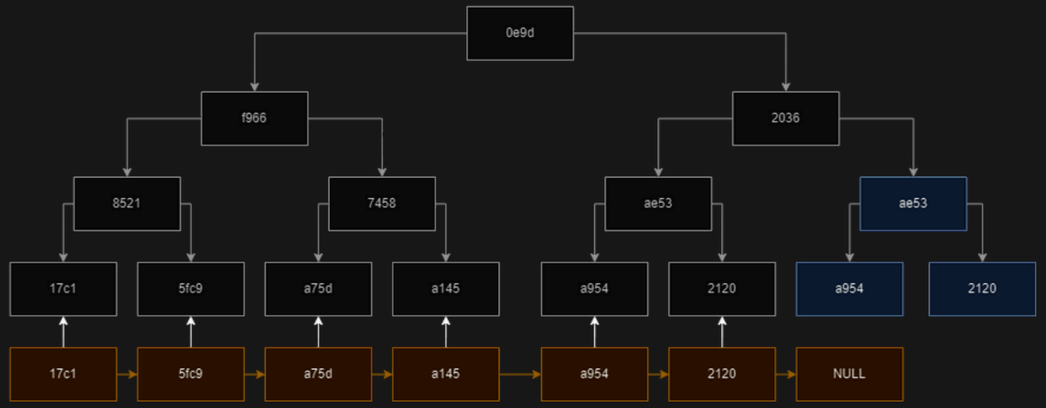
SECOND PREIMAGE ATTACK

Based on hash collisions



SECOND PREIMAGE ATTACK

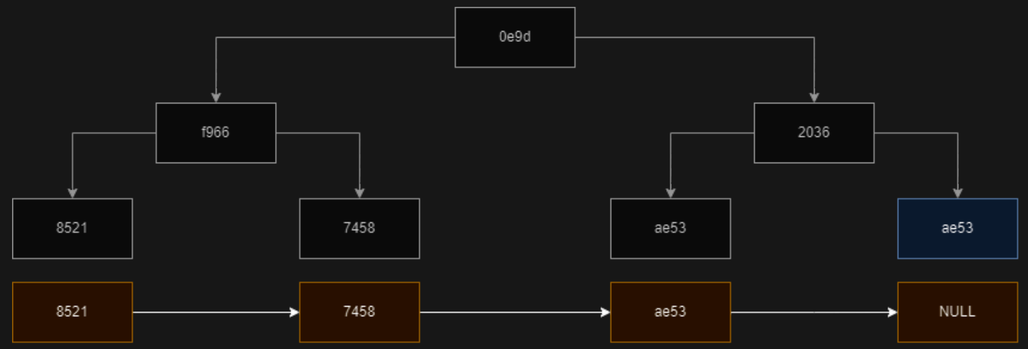
How can we ensure that?



h(M): 0e9d

size(M): 15

size(M) != size(M')



h(M'): 0e9d

size(M'): 7

COMPLEXITY

Branching factor 2 and *k*

	Average	Worst
Space	$O(n)$	$O(n)$
Search	$O(log_2(n))$	$O(log_k(n))$
Traversal	$*O(n)$	$*O(n)$
Insert	$O(log_2(n))$	$O(log_k(n))$
Delete	$O(log_2(n))$	$O(log_k(n))$
Syncronization	$O(log_2(n))$	$O(n)$

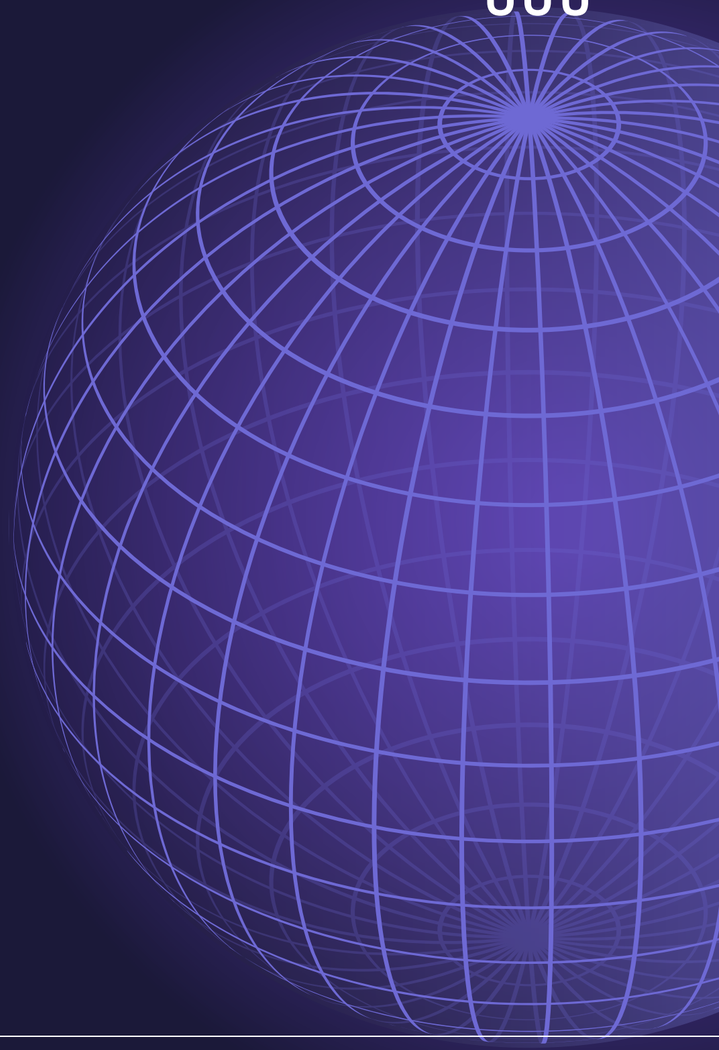
CONCLUSION

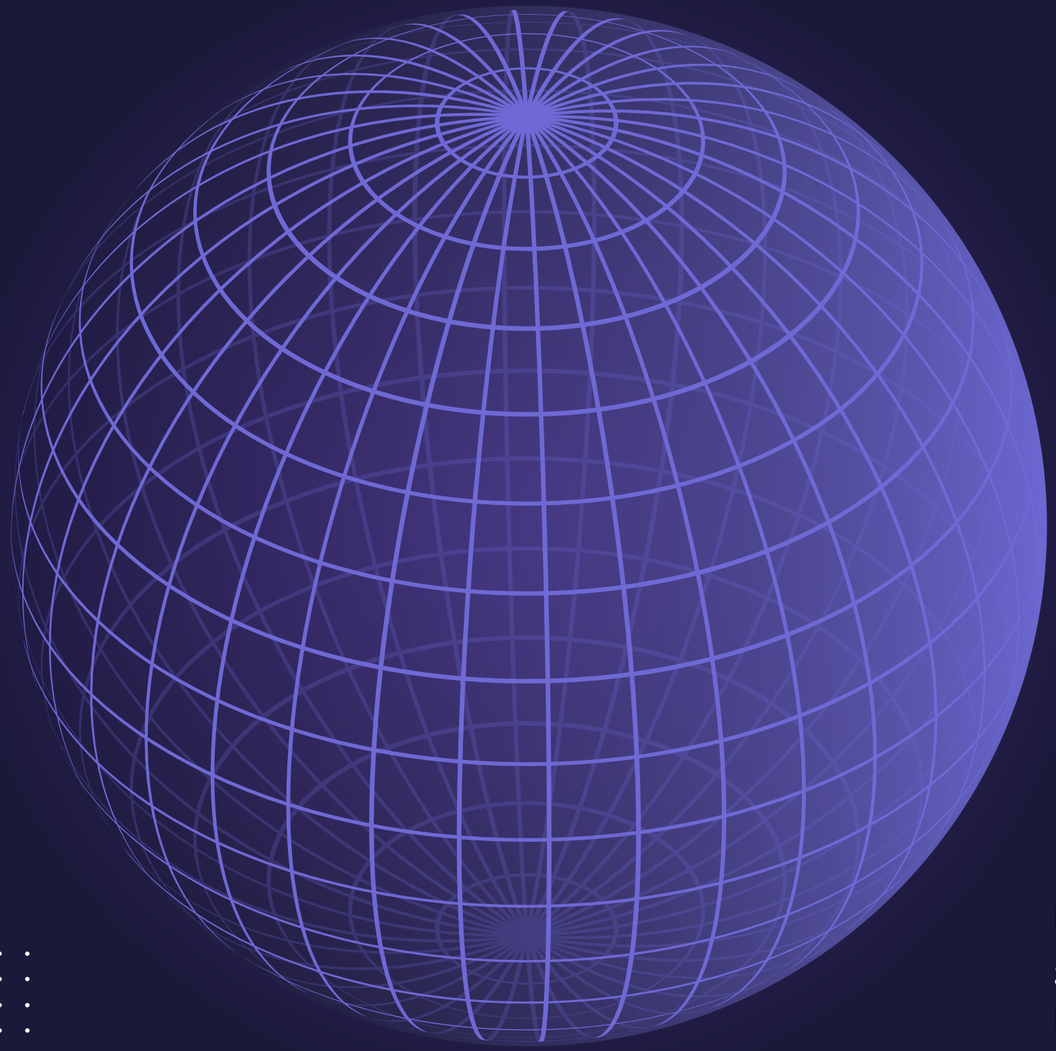
BASE OF MODERN CRYPTOGRAPHY

As we can see, Merkle trees are an essential data structure in the context of contemporary peer-to-peer technologies and cryptography systems (like blockchain). They make it possible to verify huge data structures quickly and securely while maintaining data consistency and integrity and preventing recomputation.

IMPLEMENTATION

Hands On!





THANK YOU!

Email: mauricioleite.fe@gmail.com
github: MauricioLeite
LinkedIn: mauricio-lefe