Universität Stuttgart
IPVS/AS

Using Triples as the Data Model for Blockchain Systems

Dennis Przytarski

Agenda

- Motivation
- Our approach
- Evaluation
- Example

Motivation

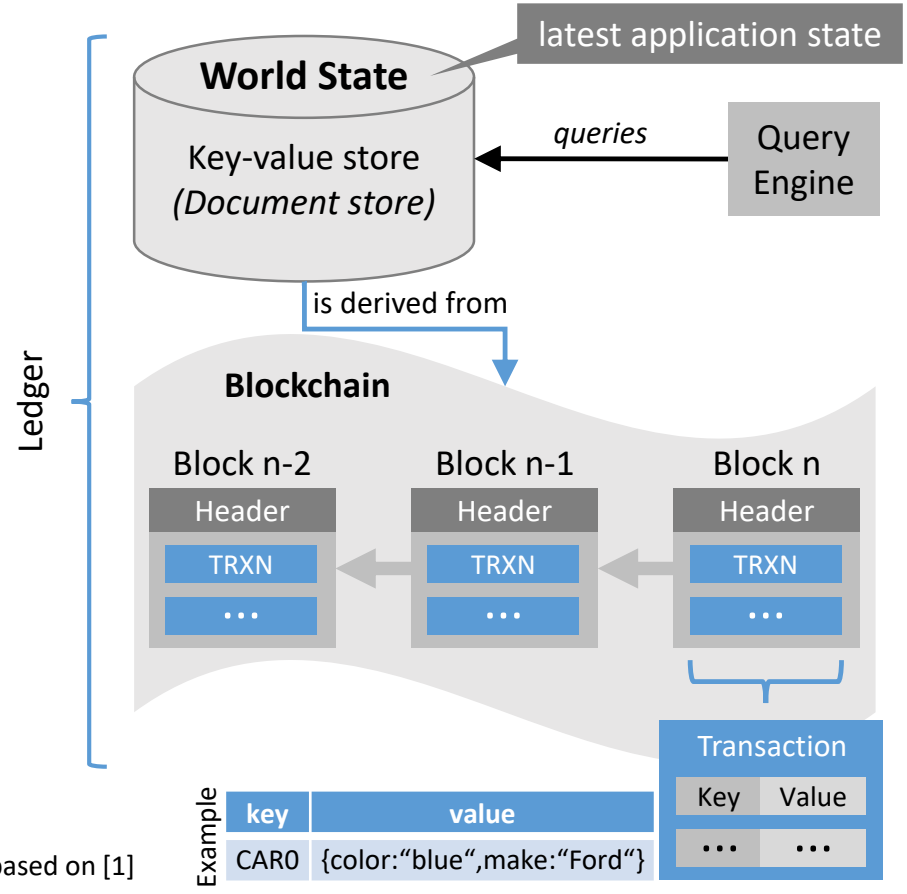
Blockchain Systems

- Initially designed for cryptocurrencies
  - Simple and fixed data model

- Other scenarios
  - Automotive
    - Full history of a vehicle
  - Real Estate
    - Record of land titles
  - Voting
    - Reduce voter fraud

Motivation

Blockchain Systems



based on [1]

- Key-value data model
- Simple query engine
  - World state
- Scenarios
  - Transportation/Trucking
    - Tracking journey stops, parcel service
  - Supply Chain Integrity
    - Food chain, waste management
- Requirements
  - Flexible information model
  - Query engine for
    - World state
    - History (analytics, audit trails)

Merge Blockchain and Database Systems

BLOCKCHAIN SYSTEMS

- immutability
- tamper-resistance

DATABASE SYSTEMS

- generic but flexible data model
- powerful query engine

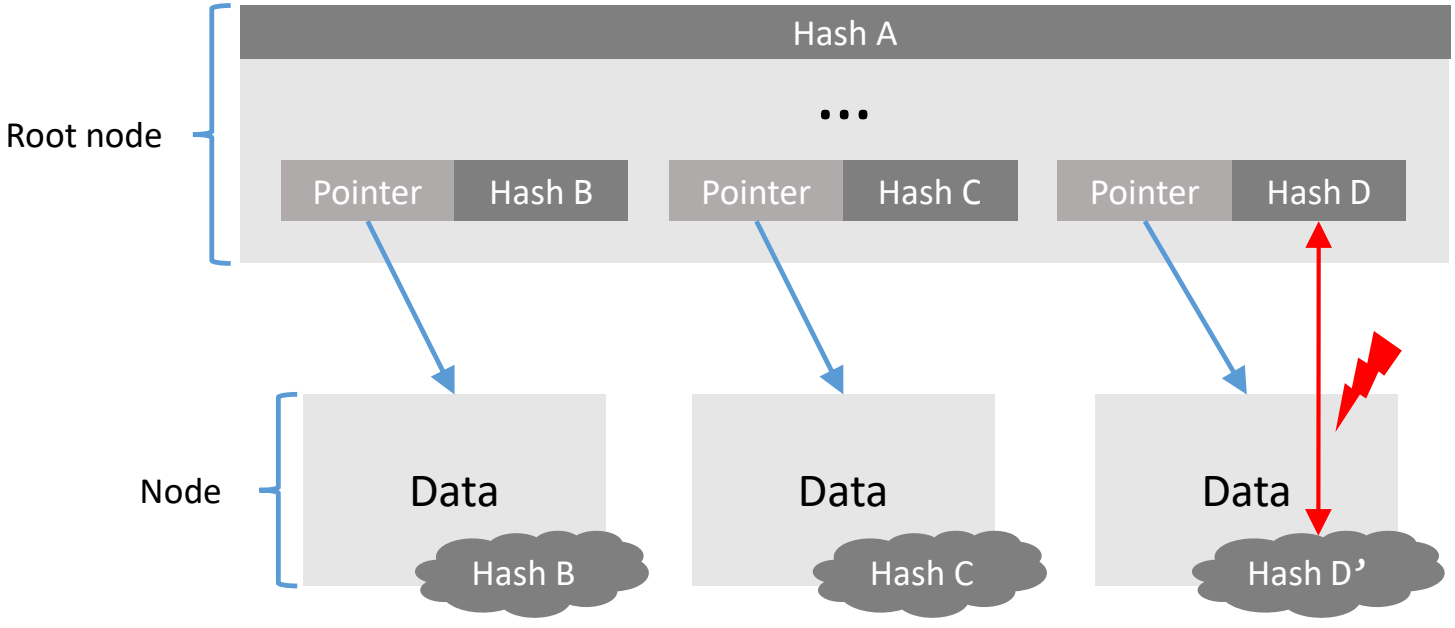
DESIGN REQUIREMENTS

- maintain a flexible information model and an efficient data representation
- support a powerful query engine
- preserve the integrity of the blockchain‘s data structure (tamper-resistance)

Data Model

- Key-value
  - values are often serialized
  - if key is not known, a full scan is necessary
  - no powerful query engine, just `get`/`set` operations
- Relational
  - too strict for immutable data
  - schemas evolve over time leading to schema changes
- Triples
  - flexible schemas without maintaining an one-size-fits-all schema
  - triples are facts → data of interest are easier to extract

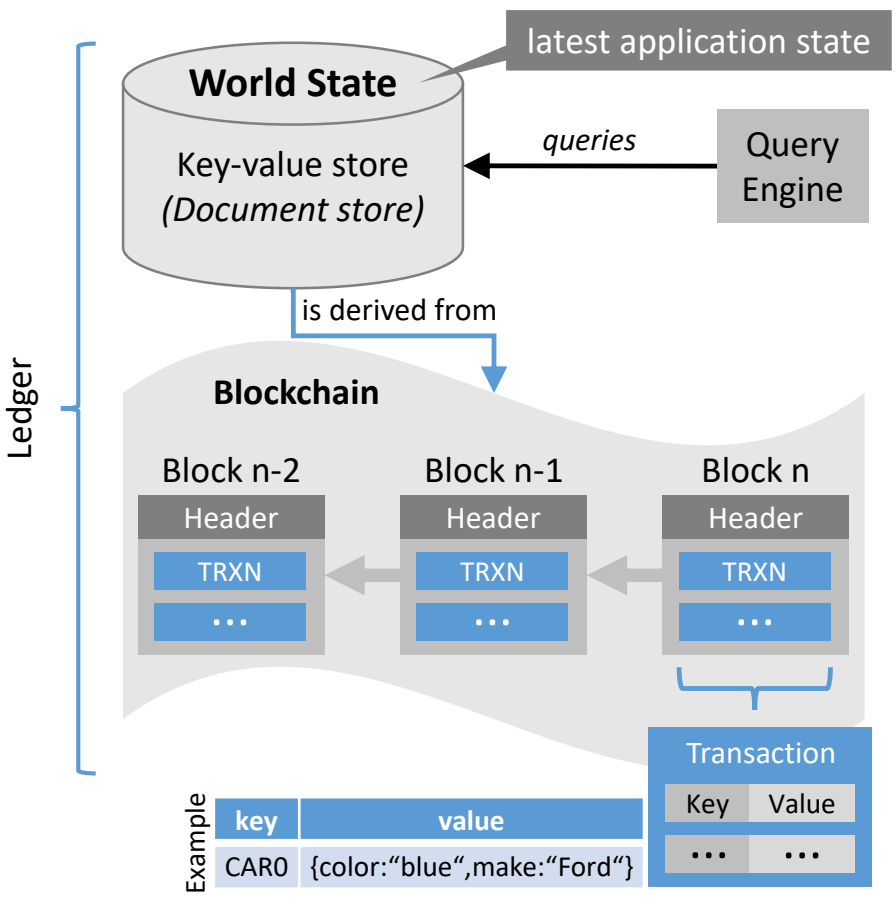
Merkle B-Tree

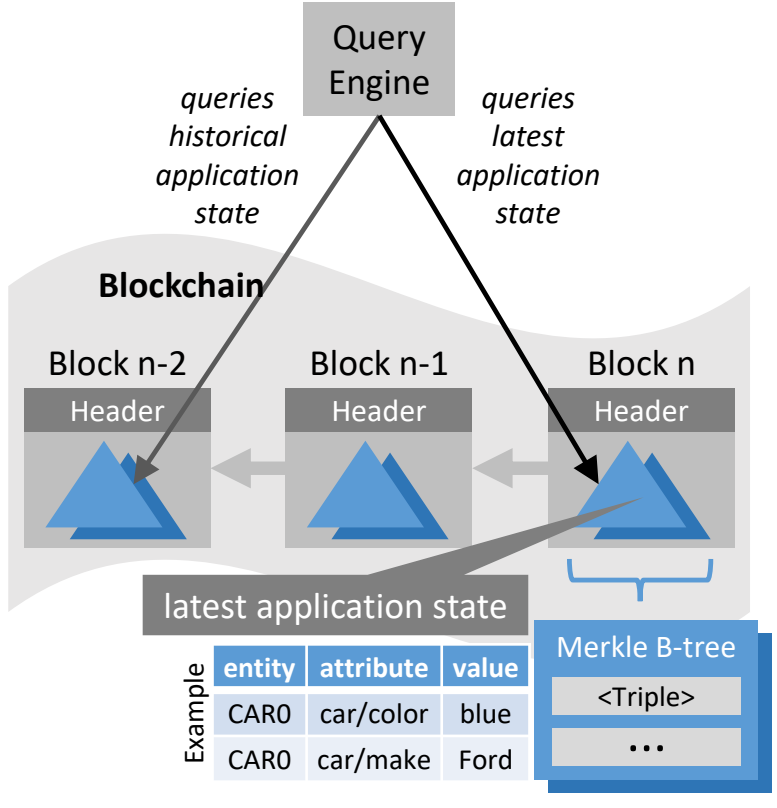


- Pointer: Memory address
- Hash: Hash over the node's data

[2]: The Merkle B-Tree: *Li, Feifei, et al. "Dynamic authenticated index structures for outsourced databases." SIGMOD 2006.*

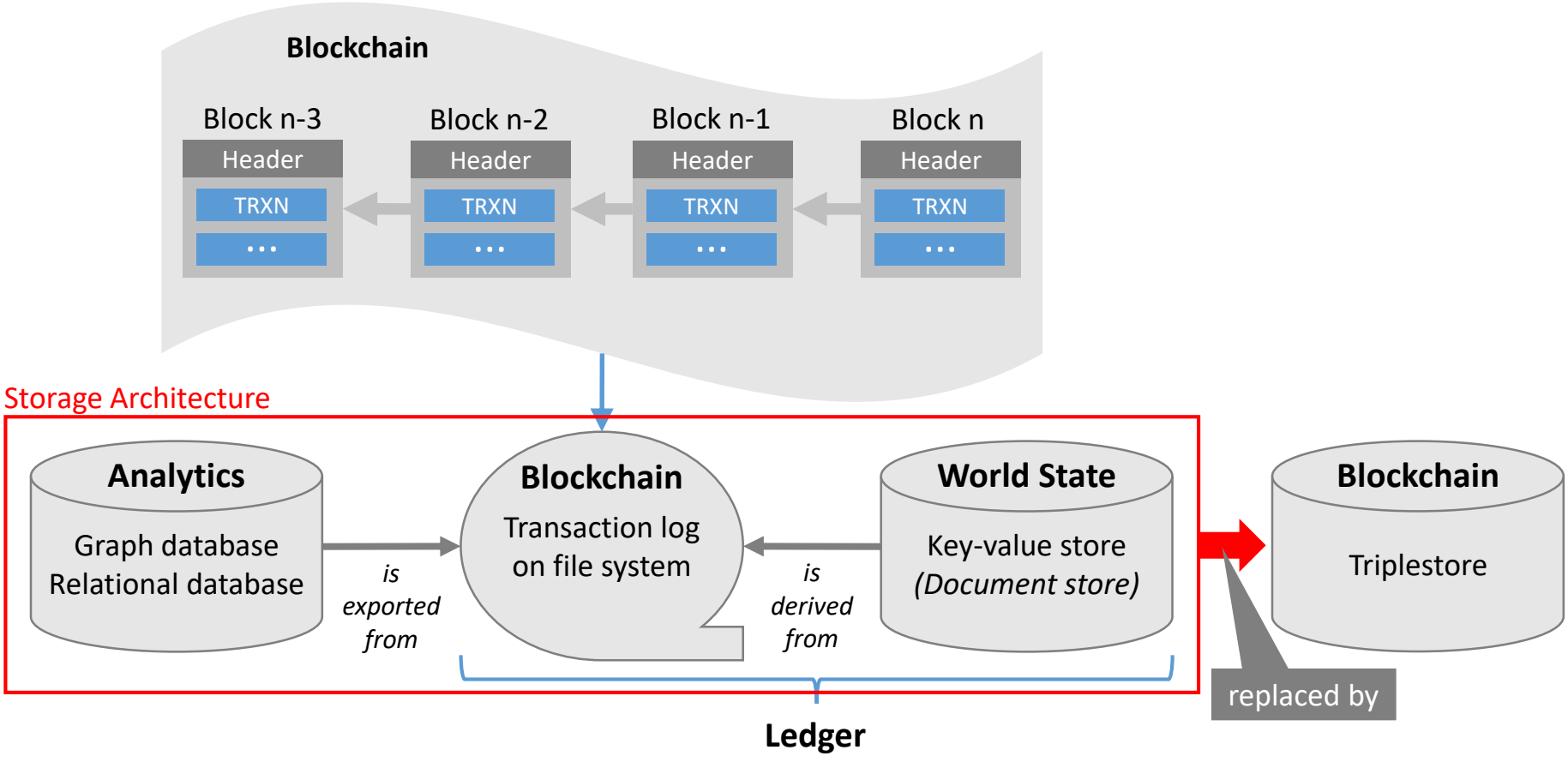
Architecture





Advantage (1/2)

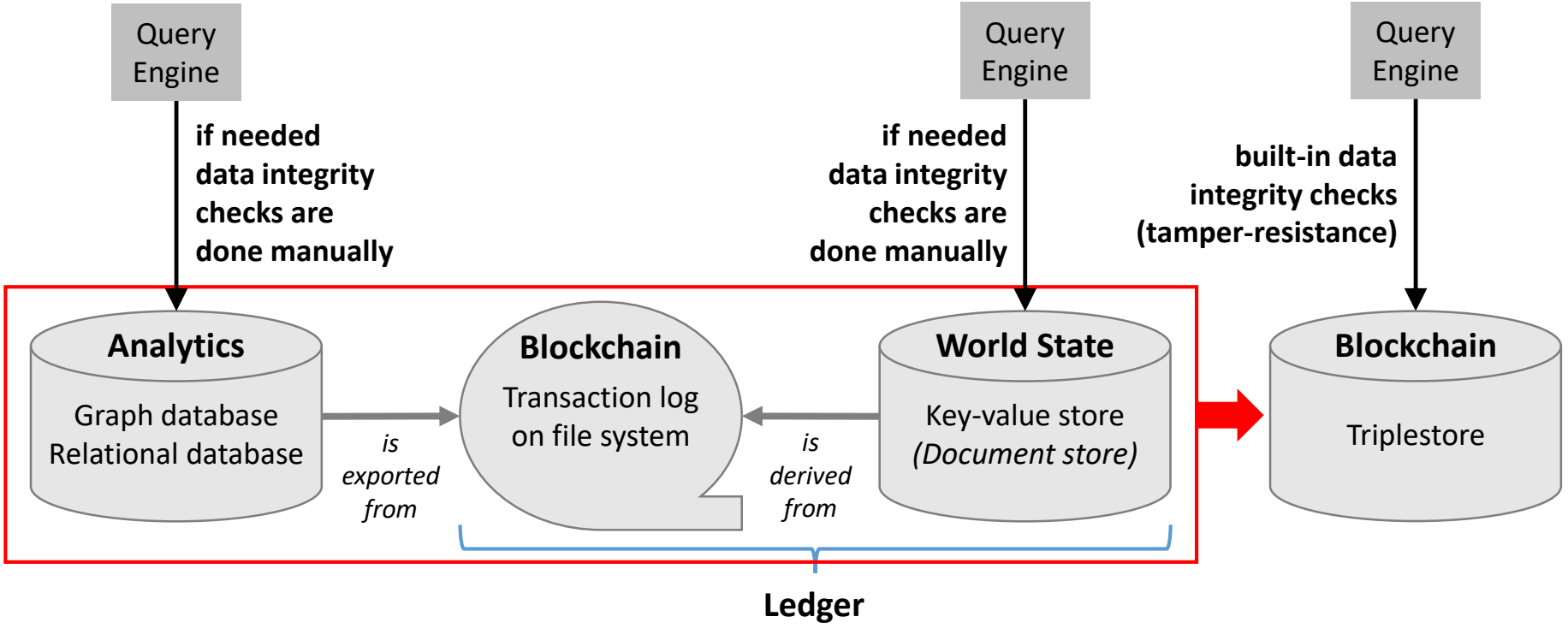
Replace three data stores by one



Advantage (2/2)

Built-in Data Integrity Checks (Tamper-resistance)

- The database systems ‚Analytics‘ and ‚World State‘ are not aware of the blockchain‘s data structure → data integrity checks must be done manually



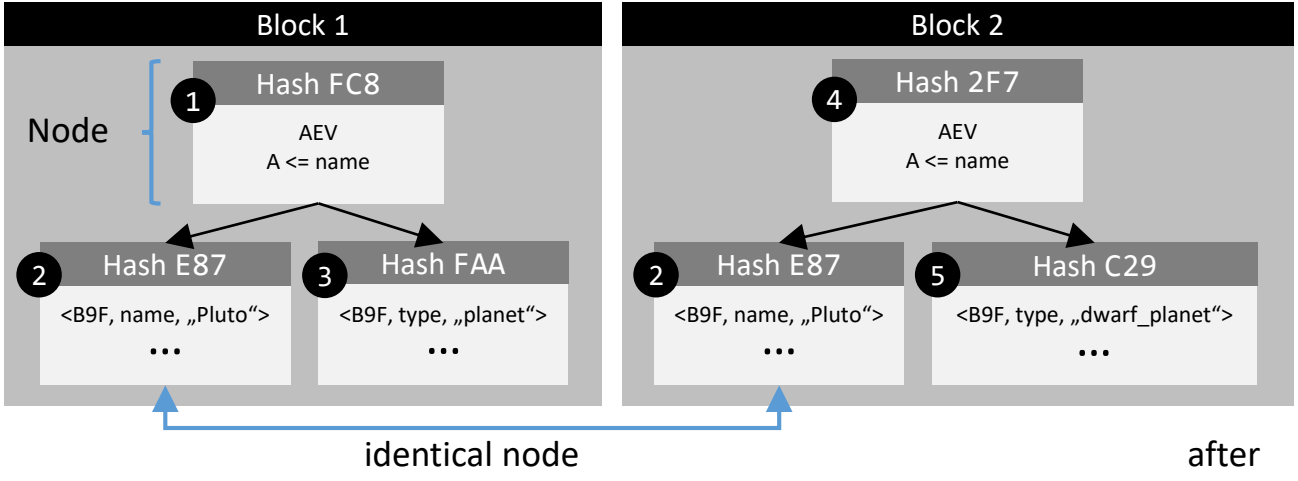
Problem

- High storage requirements
- Optimization mechanisms (→ efficient storage techniques)
  - data compression
  - data encoding
  - reuse of already stored data structures

Example

Transaction 1
<B9F, name, „Pluto“>
<B9F, type, „planet“>

Transaction 2
<B9F, type, „dwarf_planet“>



The query engine uses the Merkle B-trees of a block to compute the result of a query

Nodes are stored in a key-value store

Key	Value	
FC8	1	Root of first block
E87	2	
FAA	3	
2F7	4	Root of second block
C29	5	

after Block 1

after Block 2

Query
SELECT *?type* ASOF *n*
WHERE
[*?object* name „Pluto“]
[*?object* type *?type*]

Query result
for *n*=1: *?type* is planet
for *n*=2: *?type* is dwarf_planet

Contact Details

Thank you!

University of Stuttgart
IPVS/AS

Dennis Przytarski

Email: Dennis.Przytarski@ipvs.uni-stuttgart.de

References

- [1]: Androulaki, Elli, et al. "Hyperledger fabric: a distributed operating system for permissioned blockchains." Proceedings of the Thirteenth EuroSys Conference. ACM, 2018.
- [2]: Li, Feifei, et al. "Dynamic authenticated index structures for outsourced databases." Proceedings of the 2006 ACM SIGMOD international conference on Management of data. ACM, 2006.
- Merkle, Ralph C. "A digital signature based on a conventional encryption function." Conference on the theory and application of cryptographic techniques. Springer, Berlin, Heidelberg, 1987.
- C. Mohan, Tutorial: Blockchains and Databases (VLDB 2017)