# Survey of Blockchain in Video Streaming

Group 12

William Anwara,
Ziming Fang,
Jacob Summers

# Video Streaming Background

**Netflix**

Subscription
Video-on-Demand

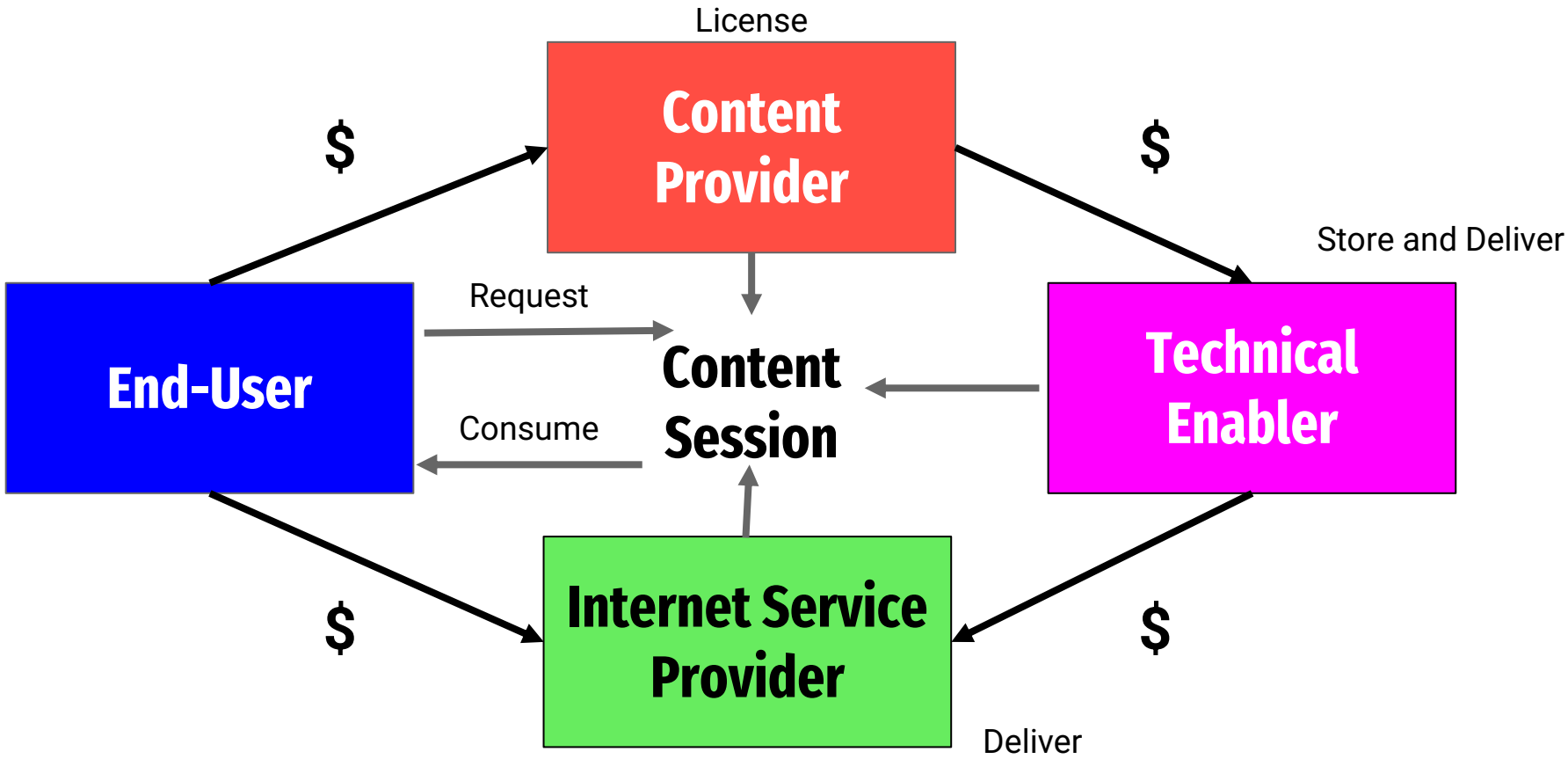
**Google Play**

Transactional
Video-on-Demand

**YouTube**

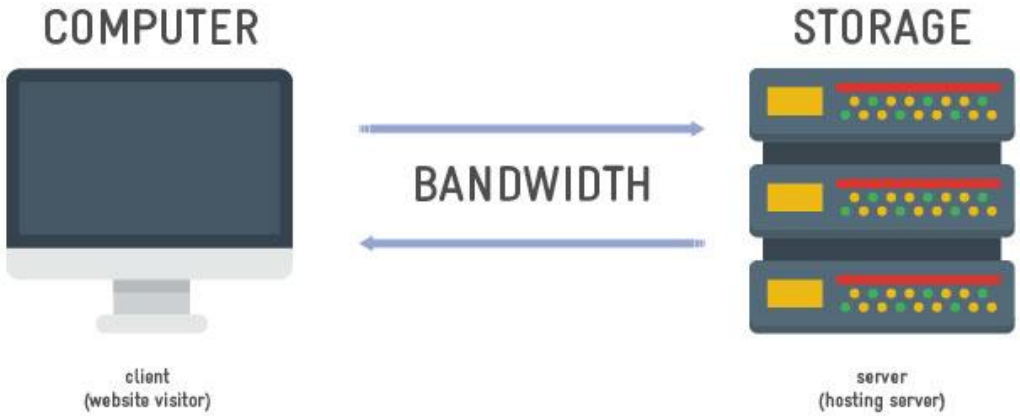
Advertising Video-
on-Demand

# Current Infrastructure



# Research Problem

- **Unhealthy stakeholder competition**
- **Systems require higher bandwidth and storage**



# Motivation

- **Increasing use of vertically integrated services**
- **Increasing streaming costs**

# Desirable Properties

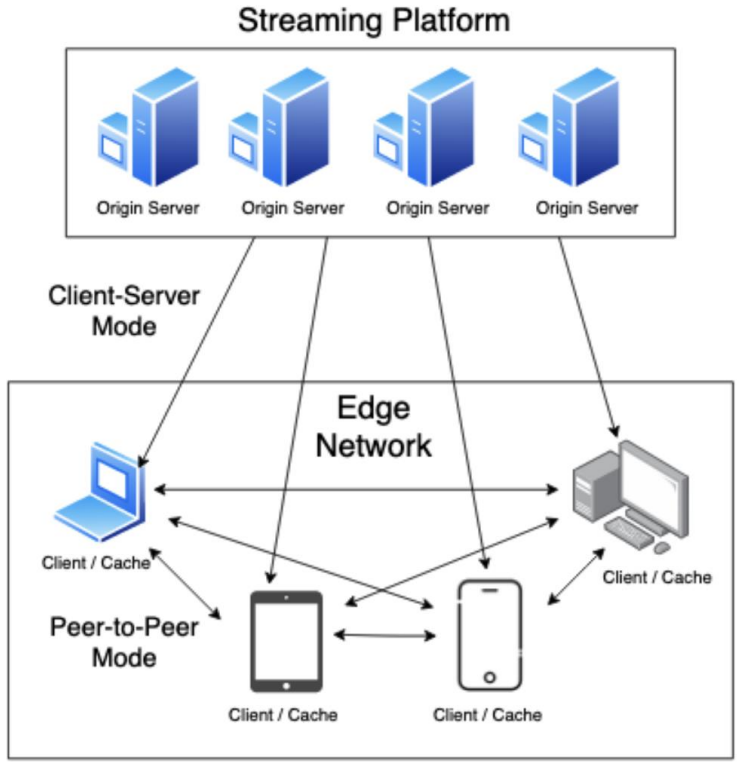
- **Healthy network of providers**
- **Reduced delivery costs**

# Proof-of-Stream

# What is Hybrid VoD Architecture

A typical hybrid VoD architecture consists of a set of **origin servers** owned by the streaming platform as well as a set of **edge devices** consisting of users and cache nodes.

User clients stream from origin servers via client-server mode whereas they stream from neighboring cache nodes in a peer-to-peer fashion.



# Research Goal

The goal of this paper

to develop a **robust incentivization mechanism** on top of CalVoD

1) Ensure an accurate view-count for each VoD

2) A modular design which can be easily integrated into existing hybrid VoD systems

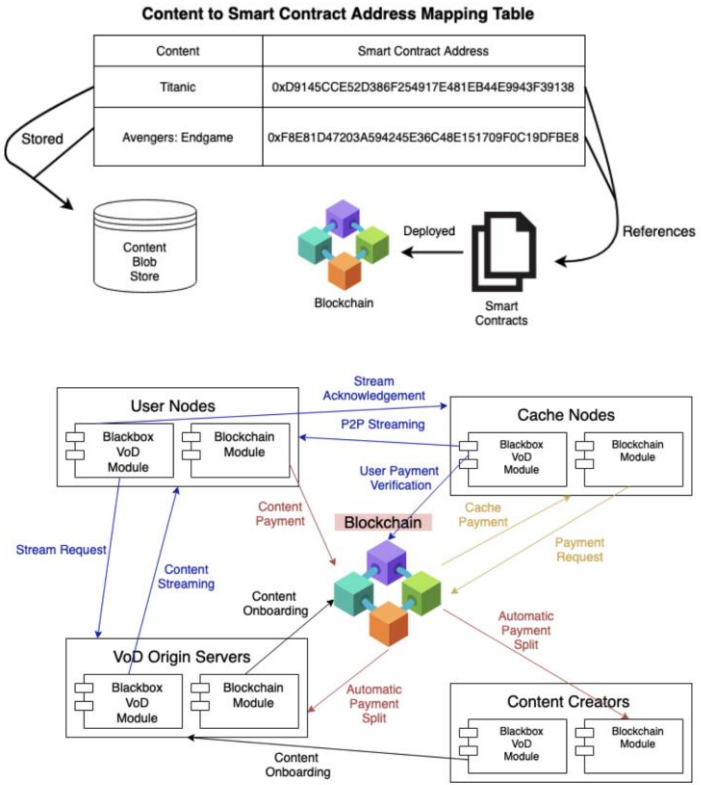



Fig. 3: System Architecture Diagram

```
contract ContentSmartContract {
    string public title;
    address payable[] public artists;
    uint256[] public artists_percentages; // NOTE:
        the index has to match those in the artists
        field
    address payable public origin_server;
    uint256 public content_price; // NOTE: this is
        in Wei, not ether, since msg.value in
        PayForStream is in Weis.
    uint256 public views;

    constructor(string memory Title, address payable
        [] memory Artists, uint256[] memory
        Artists_percentages, uint256 Content_price)
        public {
        title = Title;
        artists = Artists;
        artists_percentages = Artists_percentages;
        content_price = Content_price;
        origin_server = payable(msg.sender);
        views = 0;
    }

    function PayForStream() public payable {
        require (msg.value >= content_price);
        uint256 moneyToReturn = msg.value -
            content_price;
        payable(msg.sender).transfer(moneyToReturn);
        uint256 total = 0;
        for (uint256 i = 0; i < artists.length; i++)
            {
            uint256 current_payable = (content_price
                * artists_percentages[i]) / 100;
            total += current_payable;
            artists[i].transfer(current_payable);
        }
        origin_server.transfer(content_price - total
            );
        views += 1;
    }
}
```

Listing 1: Proof-of-Stream Smart Contract Code

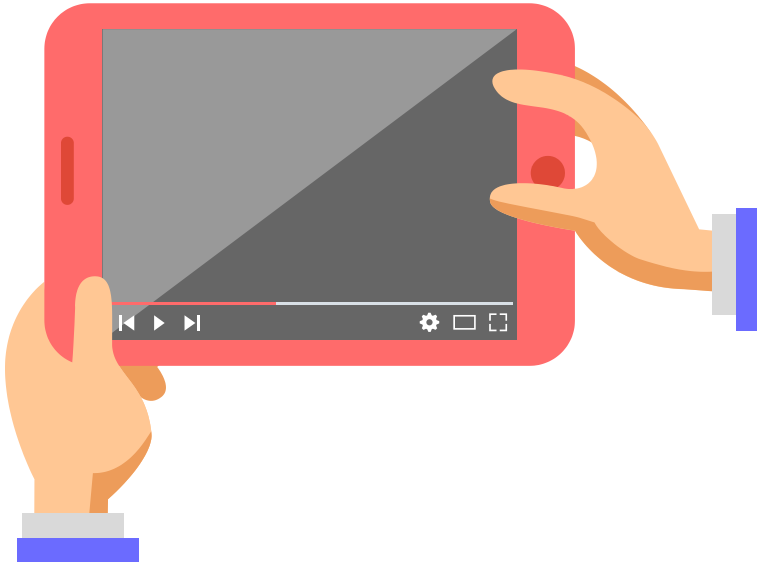
# Proposed Scheme: Proof-Of-Stream

1. **Content On-boarding**

   the process of getting a content ready for the Proof-of-Stream layer, **rather than the process of ingesting a content from an external source** into CalVoD's internal storage layer (i.e. not referring to the encoding, transcoding, decoding and storage pipeline).

# Proposed Scheme: Proof-Of-Stream

2. Content Stream Request and Payment Splitting
3. Content Streaming and Cache Acknowledgements
4. Cache Payments



# Evaluation

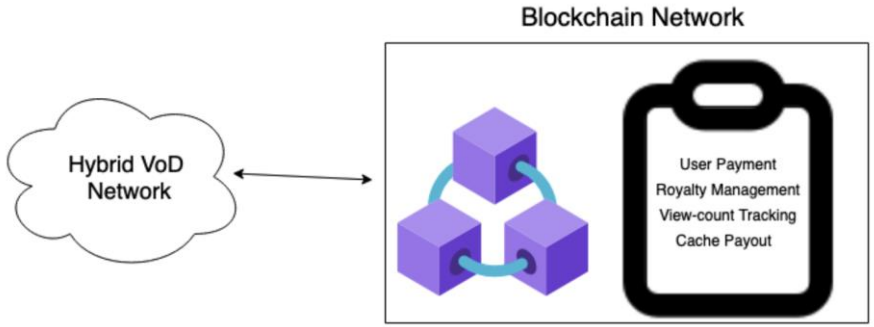
## Threat Evaluation

*1) Collusion-based Attacks*

Such attacks typically involve a colluding party of origin servers, cache nodes and users against the content creators

*2) View Inflation Attack*

The VoD platform is also incentivized to increase profit margins by overcharging advertisers.



# Evaluation

**Integration Costs**

Blockchain Modules within

- Origin Servers
  - New content smart contracts
- User Clients
  - User payments
- Partner Clients
  - Royalty-payouts for partners

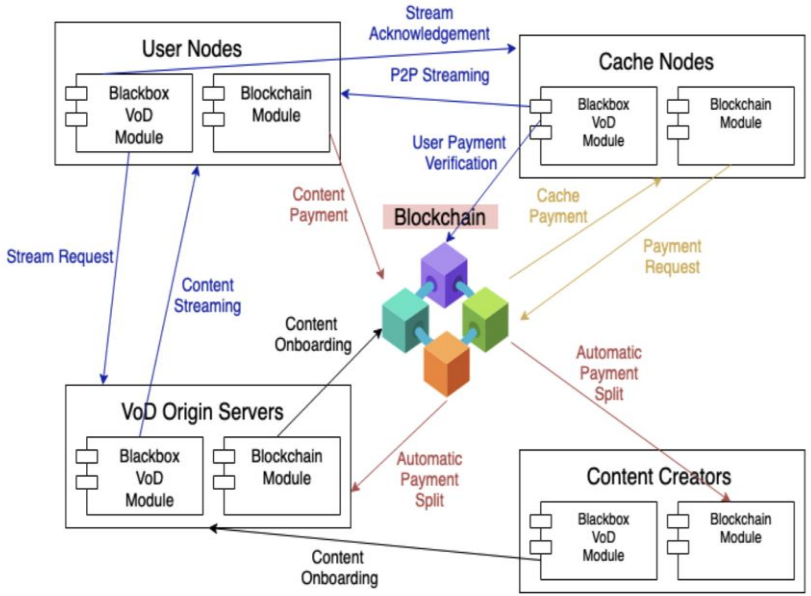



Fig. 3: System Architecture Diagram

# Evaluation

APPENDIX G. FEE SCHEDULE

The fee schedule $G$ is a tuple of 31 scalar values corresponding to the relative costs, in gas, of a number of abstract operations that a transaction may effect.

| Name | Value | Description* |
|---|---|---|
| $G_{zero}$ | 0 | Nothing paid for operations of the set $W_{zero}$. |
| $G_{base}$ | 2 | Amount of gas to pay for operations of the set $W_{base}$. |
| $G_{verylow}$ | 3 | Amount of gas to pay for operations of the set $W_{verylow}$. |
| $G_{low}$ | 5 | Amount of gas to pay for operations of the set $W_{low}$. |
| $G_{mid}$ | 8 | Amount of gas to pay for operations of the set $W_{mid}$. |
| $G_{high}$ | 10 | Amount of gas to pay for operations of the set $W_{high}$. |
| $G_{extcode}$ | 700 | Amount of gas to pay for an EXTCODESIZE operation. |
| $G_{extcodehash}$ | 700 | Amount of gas to pay for an EXTCODEHASH operation. |
| $G_{balance}$ | 700 | Amount of gas to pay for a BALANCE operation. |
| $G_{sload}$ | 800 | Paid for a SLOAD operation. |
| $G_{jumpdest}$ | 1 | Paid for a JUMPDEST operation. |
| $G_{sset}$ | 20000 | Paid for an SSTORE operation when the storage value is set to non-zero from zero. |
| $G_{sreset}$ | 5000 | Paid for an SSTORE operation when the storage value's zeroness remains unchanged or is set to zero. |
| $R_{sclear}$ | 15000 | Refund given (added into refund counter) when the storage value is set to zero from non-zero. |
| $R_{selfdestruct}$ | 24000 | Refund given (added into refund counter) for self-destructing an account. |
| $G_{selfdestruct}$ | 5000 | Amount of gas to pay for a SELFDESTRUCT operation. |
| $G_{create}$ | 32000 | Paid for a CREATE operation. |
| $G_{codedeposit}$ | 200 | Paid per byte for a CREATE operation to succeed in placing code into state. |
| $G_{call}$ | 700 | Paid for a CALL operation. |
| $G_{callvalue}$ | 9000 | Paid for a non-zero value transfer as part of the CALL operation. |
| $G_{callstipend}$ | 2300 | A stipend for the called contract subtracted from $G_{callvalue}$ for a non-zero value transfer. |
| $G_{newaccount}$ | 25000 | Paid for a CALL or SELFDESTRUCT operation which creates an account. |
| $G_{exp}$ | 10 | Partial payment for an EXP operation. |
| $G_{expbyte}$ | 50 | Partial payment when multiplied by $\lceil \log_{256}(exponent) \rceil$ for the EXP operation. |
| $G_{memory}$ | 3 | Paid for every additional word when expanding memory. |
| $G_{txcreate}$ | 32000 | Paid by all contract-creating transactions after the *Homestead* transition. |
| $G_{txdatazero}$ | 4 | Paid for every zero byte of data or code for a transaction. |
| $G_{txdatanonzero}$ | 68 | Paid for every non-zero byte of data or code for a transaction. |
| $G_{transaction}$ | 21000 | Paid for every transaction. |
| $G_{log}$ | 375 | Partial payment for a LOG operation. |
| $G_{logdata}$ | 8 | Paid for each byte in a LOG operation's data. |
| $G_{logtopic}$ | 375 | Paid for each topic of a LOG operation. |
| $G_{sha3}$ | 30 | Paid for each SHA3 operation. |
| $G_{sha3word}$ | 6 | Paid for each word (rounded up) for input data to a SHA3 operation. |
| $G_{copy}$ | 3 | Partial payment for *COPY operations, multiplied by words copied, rounded up. |
| $G_{blockhash}$ | 20 | Payment for BLOCKHASH operation. |
| $G_{quaddivisor}$ | 20 | The quadratic coefficient of the input sizes of the exponentiation-over-modulo precompiled contract. |

## Execution Costs

*• Transaction cost: This is based on the overall gas cost of sending data to the blockchain, and is typically consists for the following components:*

1) Base cost of a transaction

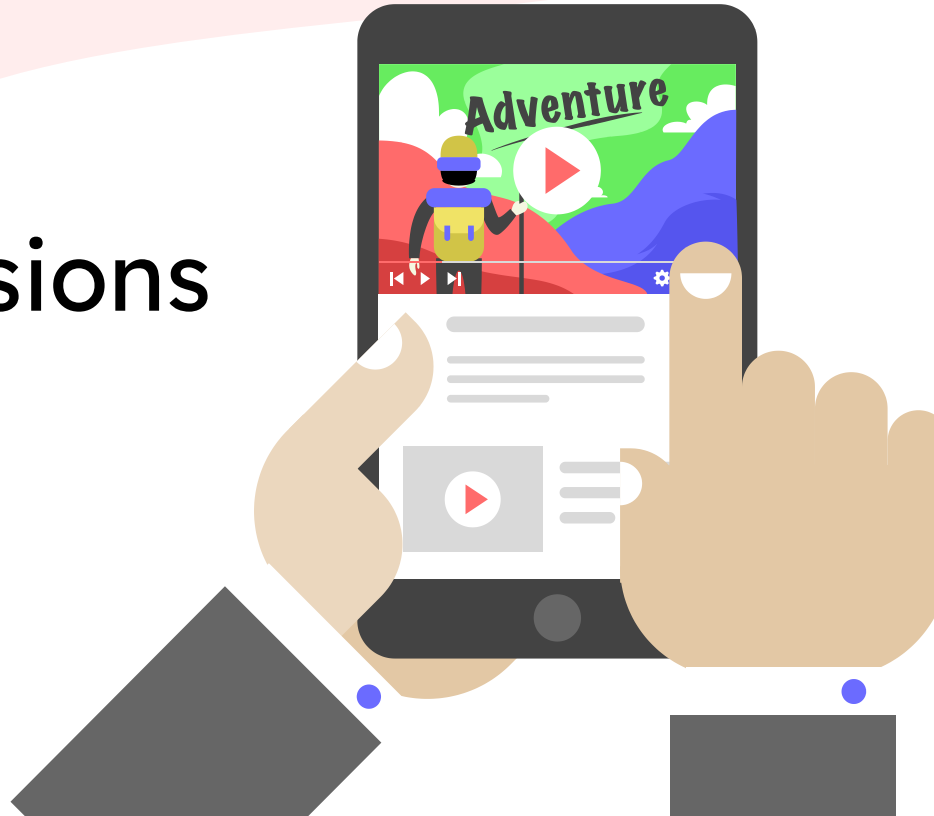
2) Cost of a contract deployment

3) Cost of every zero byte of data or code in a transaction

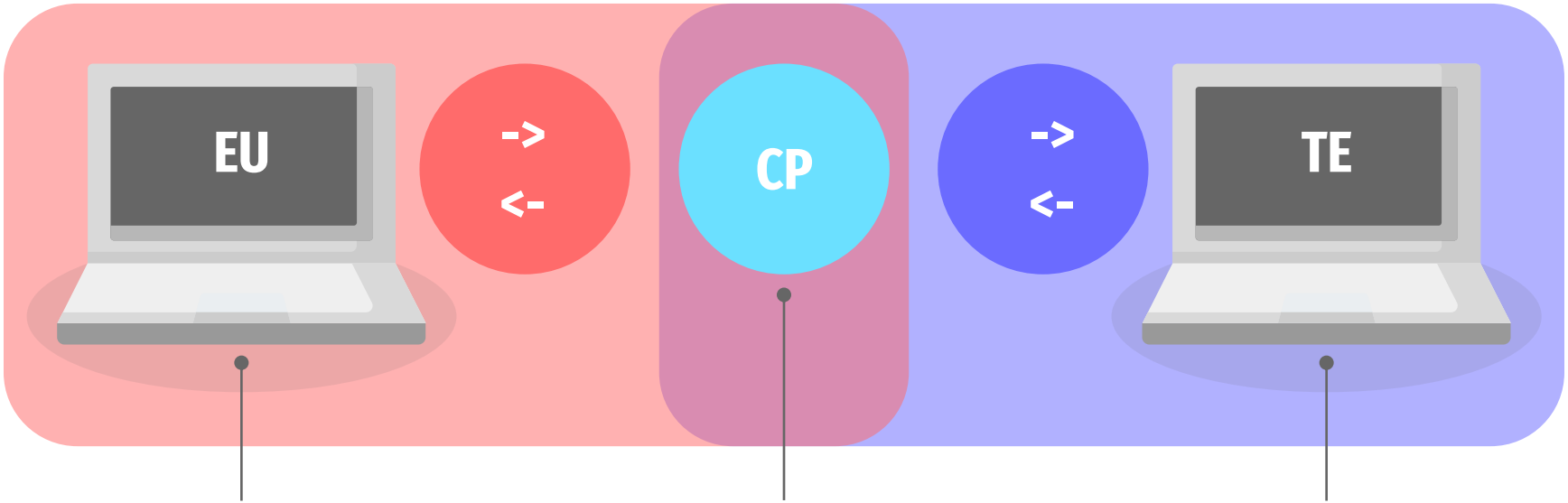
4) Cost of every non-zero byte of data or code in a transaction;

*• Execution cost: This indicates the portion of gas that is actually spent on executing the code in a transaction by the Ethereum Virtual Machine;*

# Content Sessions



# Blockchain System - Content Sessions




**End-User**

Desire for content

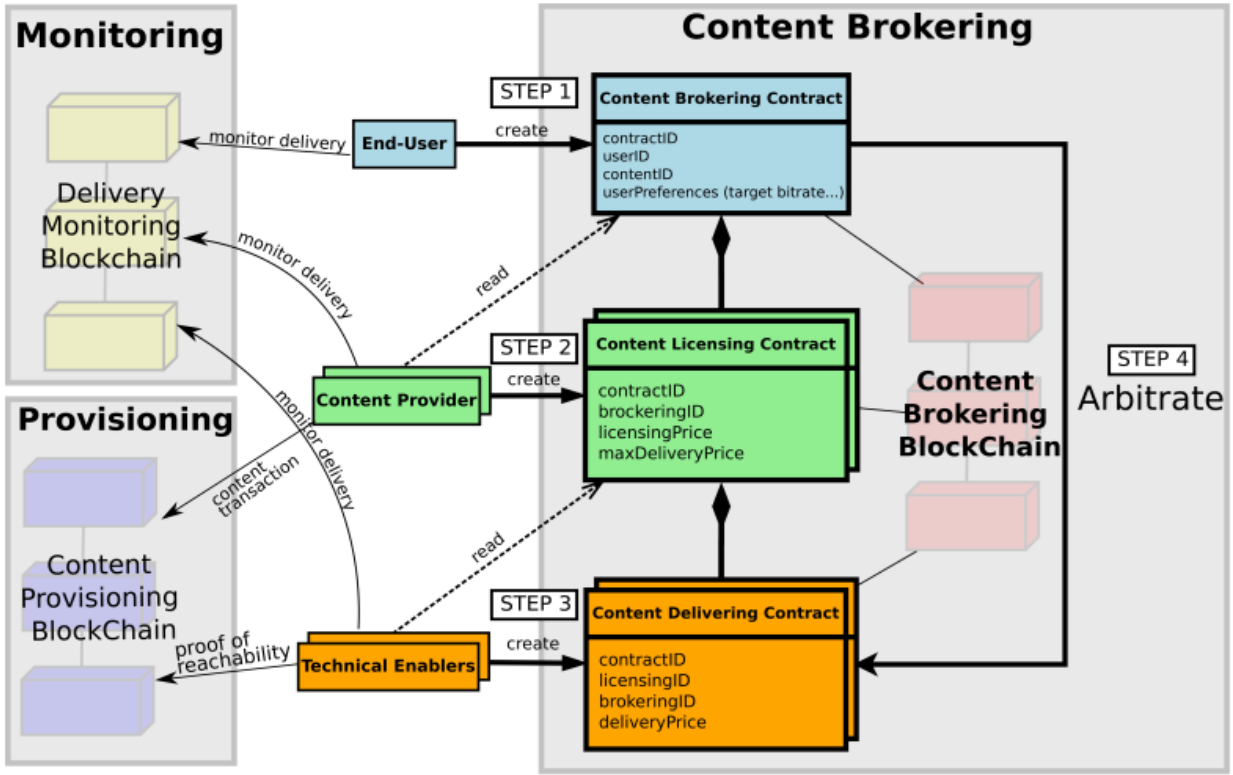
**Content Providers(CP)**

Responds to the desire with offer

**Technical Enablers (TE)**

Makes desire a reality

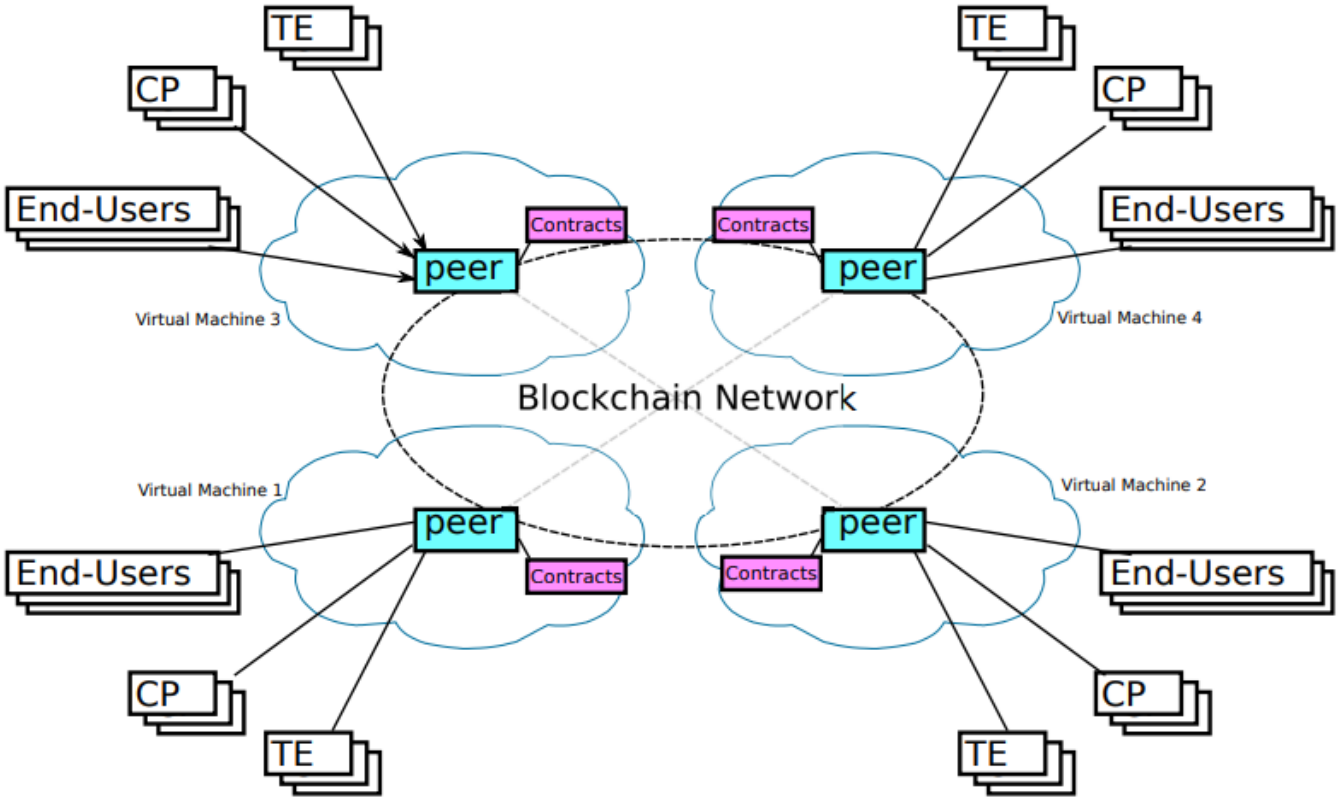
# Content Sessions System Model



# Brokering, Monitoring, and Provisioning

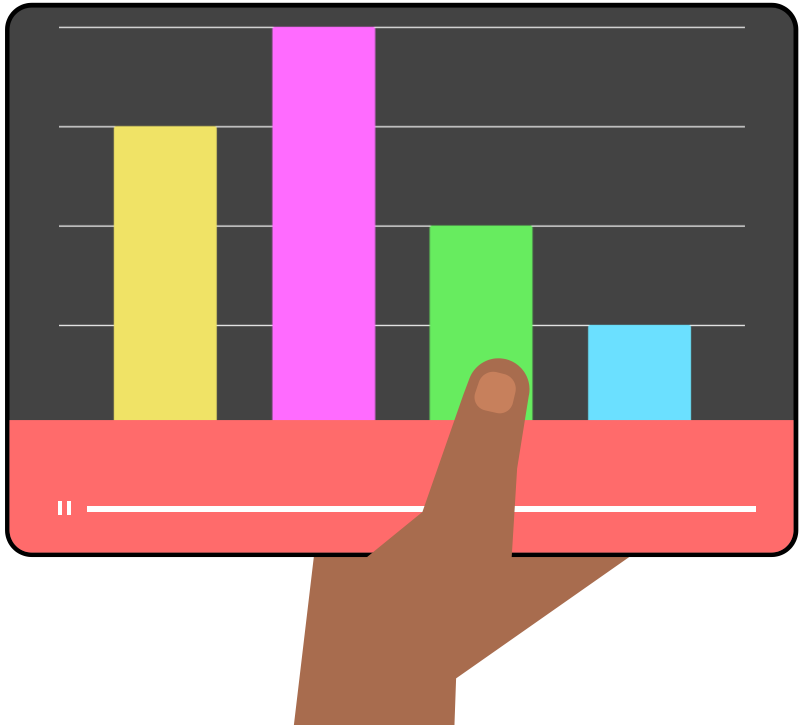
- **Content Brokering Blockchain**
  - **Content Brokering Contract**
  - **Content Licensing Contract**
  - **Content Delivery Contract**
- **Delivery Monitoring Blockchain**
- **Provisioning Blockchain**

# Network Services Chain

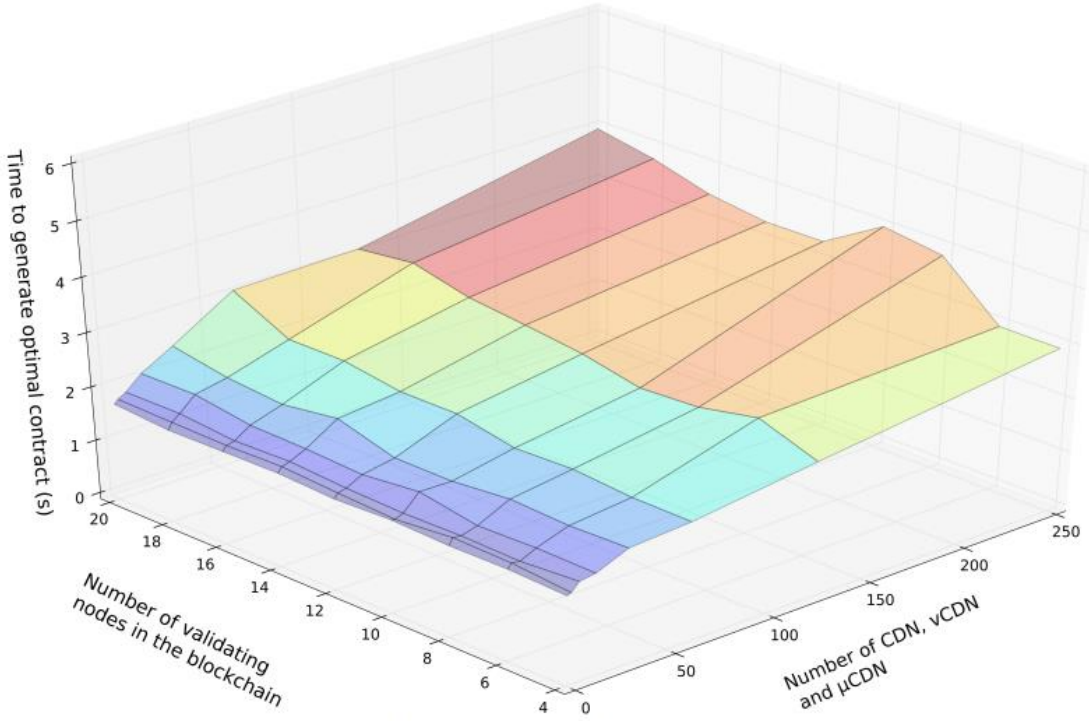


# Research Methods

- Performance = number of transactions processed per second
- Hyperledger-Fabric and Go/Java
- Looking for time to get optimal content delivery contract



# Experimental Results




(b) Performance and scalability experiment

# Proof-of-Stream and Content Session Side by Side

|  | **Proof-Of-Stream** | **Content Sessions** |
|---|---|---|
| **Architecture Type** | Client-Server + Peer-to-Peer | Client-Server |
| **Blockchain Use** | User payments, view count tracking, royalty payments, cache incentivization | Content Brokering, Delivery Monitoring, Proofs, Provisioning |

# Conclusion

The blockchain technology has a *potential role* and *advantages* for **video streaming applications**

In the presentation, we investigated and compared **2** methods to implement blockchain in video streaming

There is more work to be done to bring the technology to the marketplace successfully

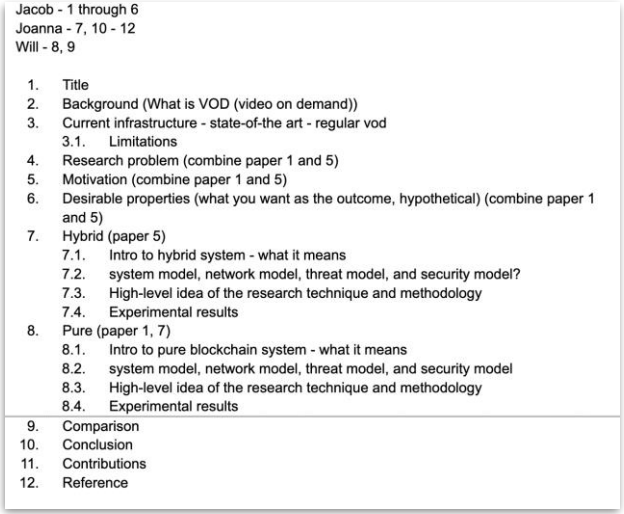
# Contributions

Team Work:

- Meeting on demand
- Made major decisions as a group
- Distributed individual works for better efficiency

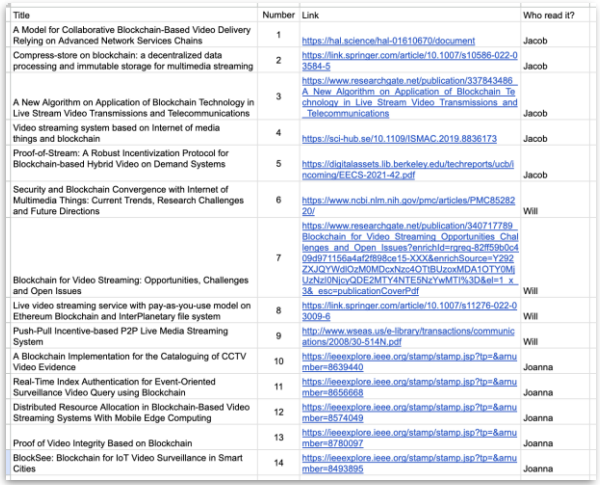
Process:

- Used Google to narrow down and choose a topic based on everyone's common interest
- Conducted extensive literature reviews to further investigate in the topic



Jacob - 1 through 6
Joanna - 7, 10 - 12
Will - 8, 9

1. Title
2. Background (What is VOD (video on demand))
3. Current infrastructure - state-of-the art - regular vod
   3.1. Limitations
4. Research problem (combine paper 1 and 5)
5. Motivation (combine paper 1 and 5)
6. Desirable properties (what you want as the outcome, hypothetical) (combine paper 1 and 5)
7. Hybrid (paper 5)
   7.1. Intro to hybrid system - what it means
   7.2. system model, network model, threat model, and security model?
   7.3. High-level idea of the research technique and methodology
   7.4. Experimental results
8. Pure (paper 1, 7)
   8.1. Intro to pure blockchain system - what it means
   8.2. system model, network model, threat model, and security model
   8.3. High-level idea of the research technique and methodology
   8.4. Experimental results
9. Comparison
10. Conclusion
11. Contributions
12. Reference



| Title | Number | Link | Who read it? |
|---|---|---|---|
| A Model for Collaborative Blockchain-Based Video Delivery Relying on Advanced Network Services Chains | 1 | https://hal.science/hal-01610670/document | Jacob |
| Compress-store on blockchain: a decentralized data processing and immutable storage for multimedia streaming | 2 | https://link.springer.com/article/10.1007/s10586-022-03584-5 | Jacob |
| A New Algorithm on Application of Blockchain Technology in Live Stream Video Transmissions and Telecommunications | 3 | https://www.researchgate.net/publication/337843486_A_New_Algorithm_on_Application_of_Blockchain_Technology_in_Live_Stream_Video_Transmissions_and_Telecommunications | Jacob |
| Video streaming system based on Internet of media things and blockchain | 4 | https://sci-hub.se/10.1109/ISMAC.2019.8836173 | Jacob |
| Proof-of-Stream: A Robust Incentivization Protocol for Blockchain-based Hybrid Video on Demand Systems | 5 | https://digitalassets.lib.berkeley.edu/techreports/ucb/incoming/EECS-2021-42.pdf | Jacob |
| Security and Blockchain Convergence with Internet of Multimedia Things: Current Trends, Research Challenges and Future Directions | 6 | https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8528220/ | Will |
| Blockchain for Video Streaming: Opportunities, Challenges and Open Issues | 7 | https://www.researchgate.net/publication/340717789_Blockchain_for_Video_Streaming_Opportunities_Challenges_and_Open_Issues?enrichId=rgreq-82ff59b0c409d971156a4af2f898ce15-XXX&enrichSource=Y292ZXJQYWdlOzM0MDcxNzc4OTtBUzoxMDA1OTY0MjUzNzI0NjcyQDE2MTY4NTE5NzYwMTI%3D&el=1_x_3&_esc=publicationCoverPdf | Will |
| Live video streaming service with pay-as-you-use model on Ethereum Blockchain and InterPlanetary file system | 8 | https://link.springer.com/article/10.1007/s11276-022-03009-6 | Will |
| Push-Pull Incentive-based P2P Live Media Streaming System | 9 | http://www.wseas.us/e-library/transactions/communications/2008/30-514N.pdf | Will |
| A Blockchain Implementation for the Cataloguing of CCTV Video Evidence | 10 | https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8639440 | Joanna |
| Real-Time Index Authentication for Event-Oriented Surveillance Video Query using Blockchain | 11 | https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8656668 | Joanna |
| Distributed Resource Allocation in Blockchain-Based Video Streaming Systems With Mobile Edge Computing | 12 | https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8574049 | Joanna |
| Proof of Video Integrity Based on Blockchain | 13 | https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8780097 | Joanna |
| BlockSee: Blockchain for IoT Video Surveillance in Smart Cities | 14 | https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8493895 | Joanna |

# Implementation References

N. Herbaut and N. Negru, “A Model for Collaborative Blockchain-Based Video Delivery Relying on Advanced Network Services Chains,” *IEEE Communications Magazine*, vol. 55, no. 9, pp. 70–76, Jun. 2021. Available: https://hal.science/hal-01610670/document

Y. Tan, S. Kadhe, and K. Ramchandran, “Proof-of-Stream: A Robust Incentivization Protocol for Blockchain-based Hybrid Video on Demand Systems,” 2021. Accessed: Apr. 24, 2023. [Online]. Available: https://digitalassets.lib.berkeley.edu/techreports/ucb/incoming/EECS-2021-42.pdf

# Thanks

Credit: Template provided by Slidesgo