at&t

Loss-tolerant Real-time Content Integrity Validation for P2P Video Streaming

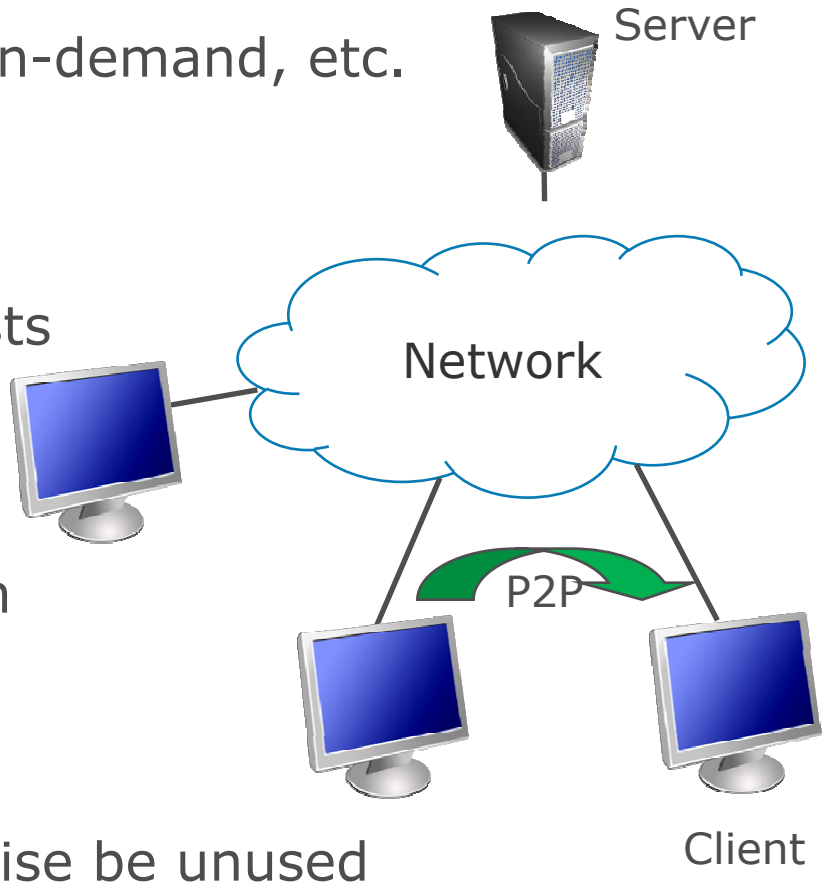
Fang Yu*, Vijay Gopalakrishnan+, K.K. Ramakrishnan+, and David Lee*

*The Ohio State University, +AT&T Labs - Research

Content Delivery

Content Delivery has become and will continue to be a major role of networks

- Includes IPTV, file sharing, video-on-demand, etc.



Downside: expensive to providers

- In terms of server & bandwidth costs

P2P is an attractive alternative

- Reduces server load and bandwidth requirement
- Increases system capacity
- Utilizes resource that might otherwise be unused
- Potentially lowers network cost and delay for clients

Ensuring Integrity of Content

Using peers poses threats to Content Integrity

- Peers can modify content out of malice or otherwise
- Peers may send random "junk" or inappropriate content

Unacceptable, esp. for paid content delivery service

- Bad quality-of-experience
- Negative publicity

→ Loss of business

Difficult for provider to ensure content validity in a pure P2P environment

- No view into what was transferred from a peer

Need capability to validate content from peers

Videos: What can a malicious peer do?

Context: Video streaming in a P2P environment

Transfer wrong chunks of a video to a client

Re-encode video

Insert or Overlay images

- For profit (e.g., Ads)
- Disrepute to the service

Introduce glitches: Degrade quality of the viewing experience

- By randomly corrupting packets

Challenges in Validating Video Content

Real-time: Data downloaded is "immediately" displayed

- Validation has to be performed in real-time (done **on-the-fly**)

Sensitive: Small changes can lead to noticeable disruption

- Checking at coarse-granularity (e.g., multiple seconds) may be too late to prevent video disruption → must validate at **small granularity**

Loss-tolerant: UDP is often the transport

- Decoders are built to handle loss of a few packets → need to be equally **resilient to packet loss**

Lightweight: Video streaming is processor/bandwidth intensive

- **Low computation and communication overheads**

All this while adequately assuring content integrity

Outline

- Introduction
- **Cooperative Peer Assists and Multicast (CPM)**
- Proposed Approach
- Evaluation
- Conclusion

Video-on-Demand using CPM

Unified approach for efficient and scalable VoD using

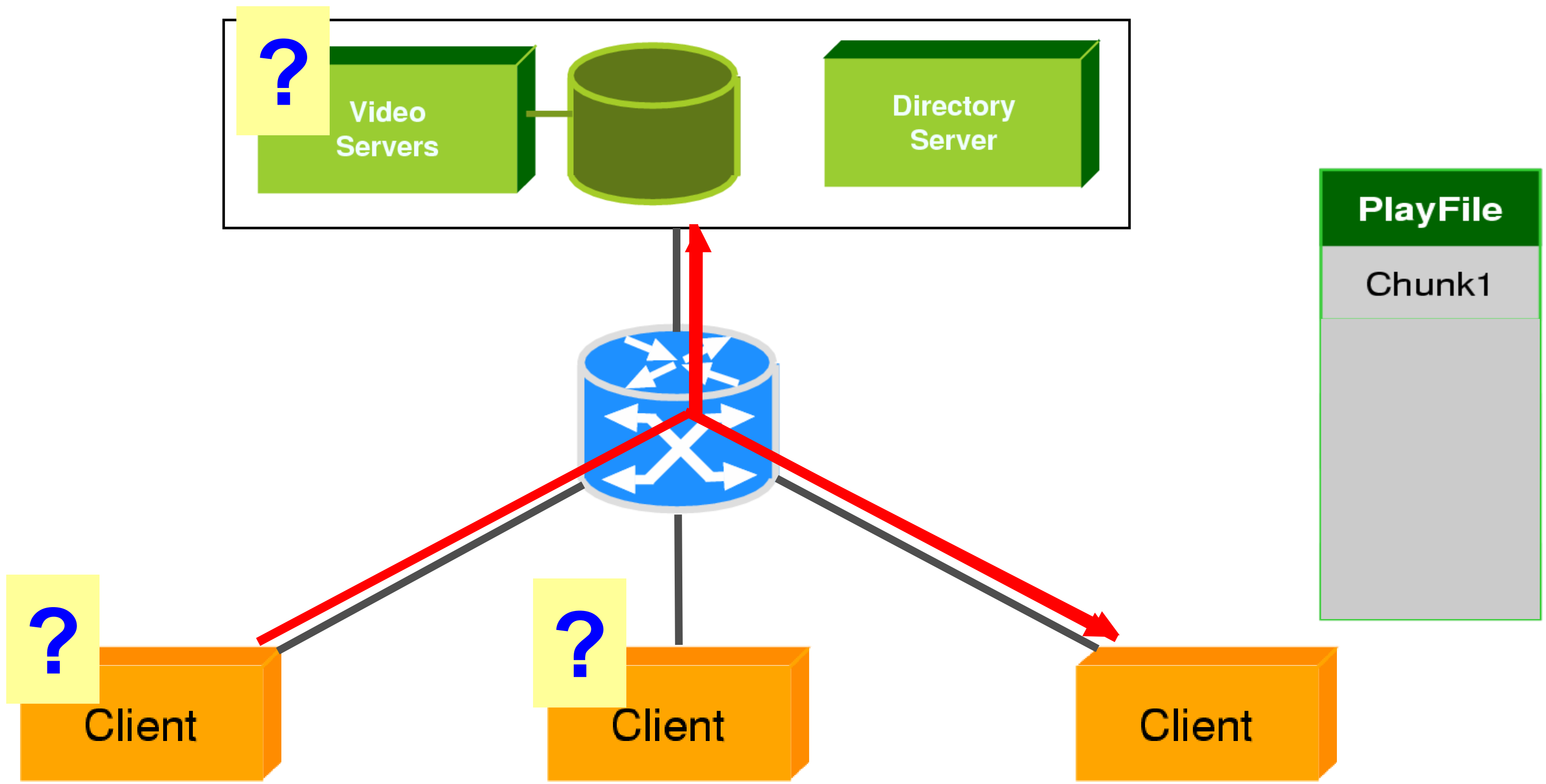
- Caching at clients (eliminates startup time and transfer cost)
- Multicast from server (resource efficient)
- Localized P2P data transfer (reduces network cost)
- Unicast from server (for rarely requested videos)

CPM assumes that a video consists of a sequence of chunks

- Chunk (~30 sec) is the smallest addressable unit
- All interactions happen at the chunk level

Goal is to be adaptive and flexible, achieved by dynamically identify best *source* and *mechanism* to deliver content

CPM: How does it work?



Client dynamically identifies best source
for each chunk in order
Client requests for a video
chunks in the video

Outline

- Introduction
- Cooperative Peer Assists and Multicast (CPM)
- **Proposed Approach for Validation**
- Evaluation
- Conclusion

Proposed Solution

Packet-level (pseudo-)random sampling for validation

- Packet-level → Right granularity, on-the-fly validation, loss-tolerant (lost packet not validated)
- Random → High security, hard for malicious peer to guess the selected set then evade
- Sampling → Low computation and communication overheads

"Ground truth": Secure one-way hash (e.g., MD5) from server

- Secure one-way hash: Common effective method (e.g., software updates, file downloads)
- From server: Leverage the trust between server and clients

Server Operations

Preprocessing: Computes and stores hashes for all packets in each video

- Hash for each packet computed only once
- Predefined packet size, oblivious to data format

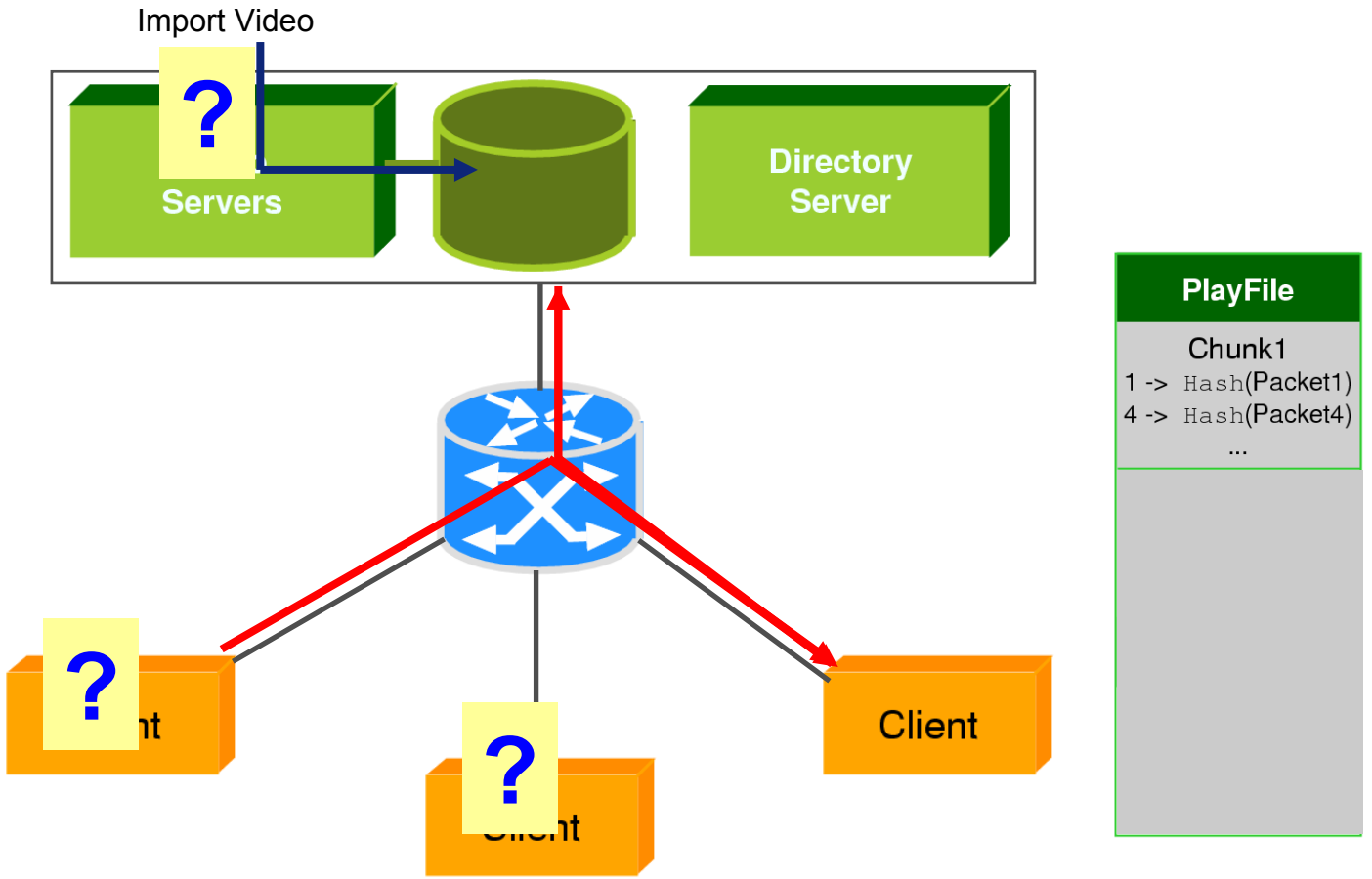
Video delivery: Upon request from client for a video

1) Generates a unique "playfile" that includes
   - Chunk IDs
   - Sequence numbers of packets (pseudo-)randomly selected for validation
     - Selection is customizable (based on video type, customer profile, playback device resolution, video bitrate, number of malicious peers in the system, etc.)
   - Correct hashes (from preprocessing) of selected packets
2) Securely sends the "playfile" to client

Client Operations

1) Request video from server
2) Receive the dynamically generated "playfile"
3) Client performs validation: For packet received from peer that is marked for validation in the playfile
   i. Compute its hash
   ii. Compare resulting hash against the correct hash in the playfile
      - If different, packet is invalid and not played
      - Otherwise valid and forward to the player
   iii. Keep a count of number of corrupted packets received from each peer
      - When count hits a threshold *T* (configurable), abort transfer and seek alternative source; peer is locally blacklisted as a corrupting peer
4) Report chunk corrupting peer (if any) to Directory Server

Overall Process



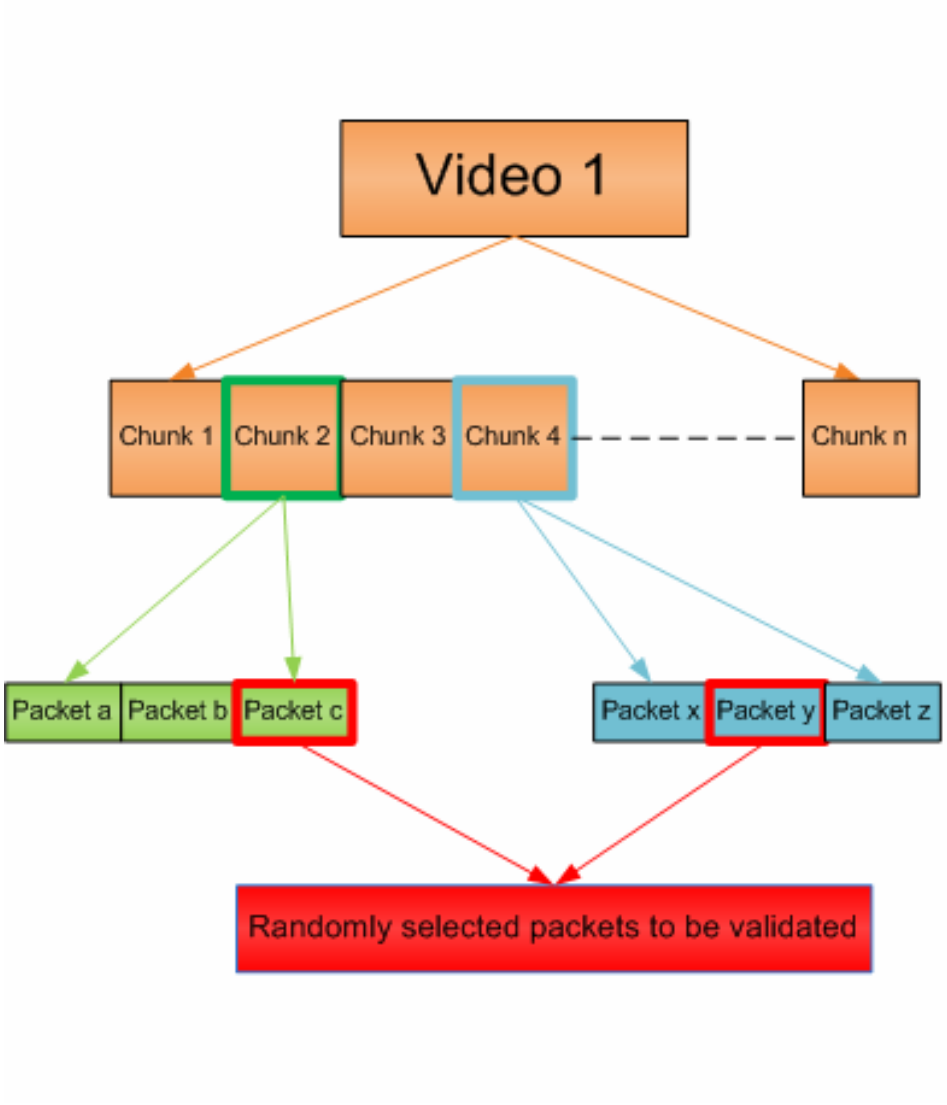
Client requests video, gets playfile

Server responds with a playfile including packets hashes

Chunkify video, get hashes of packets in chunk

Client verifies playfile by computing hash of each packet, ordered in the playfile

Random Packet Sampling

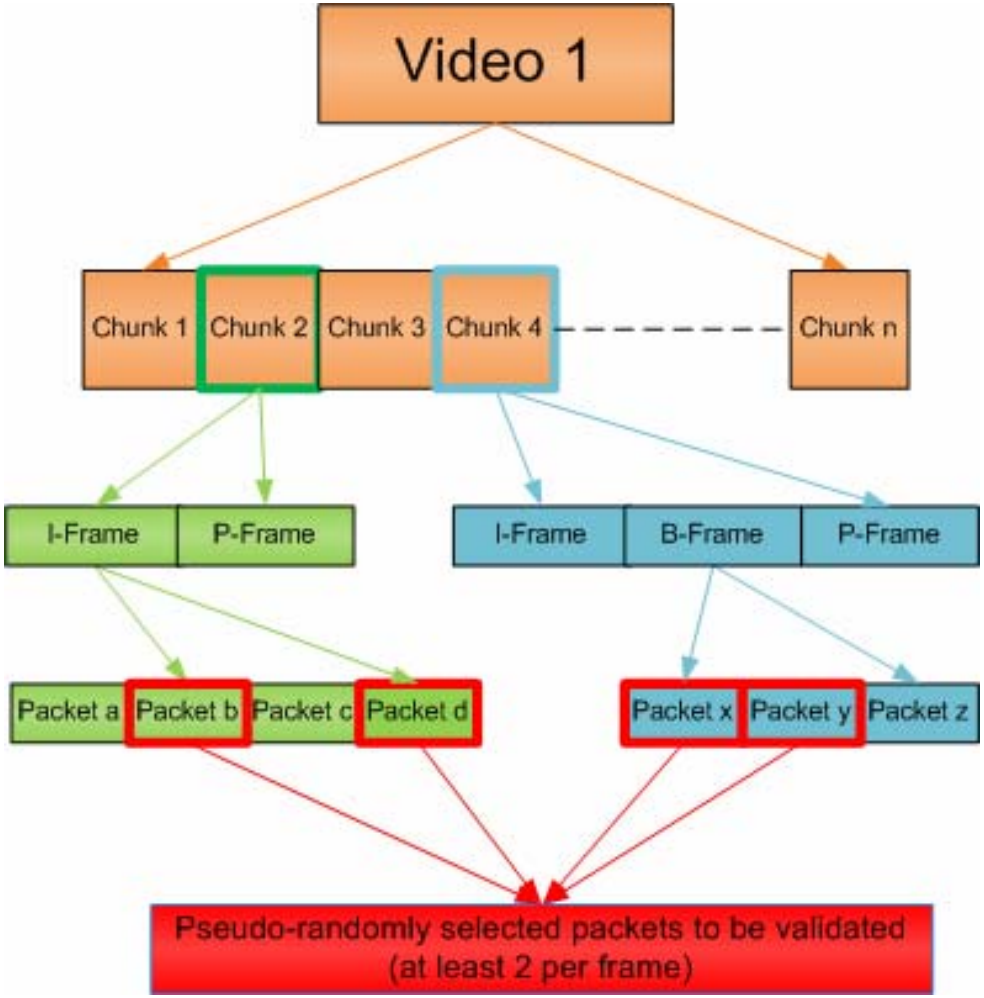


Playfile for Video 1
(Unique for each client who requests Video 1)

IDs for chunks in this video
Chunk 1
Packets to be validated
Packet ...
...
Chunk 2
Packets to be validated
Packet c, Hash (Packet c)
...
Chunk 3
Packets to be validated
Packet ...
...
Chunk 4
Packets to be validated
Packet y, Hash (Packet y)
....
...
...
...
Chunk n
Packets to be validated
Packet ...
...

Pseudo-Random Packet Sampling

- Pseudo-random: Select at least 2 packets per frame
- At least 2 to be confident that corruption is not due to faulty hardware
- Target attacks that modify only specific frames (e.g., re-encode content, add overlay etc.)
- 100% detection for such attacks



Outline

- Introduction
- Cooperative Peer Assists and Multicast (CPM)
- Proposed Approach
- **Evaluation**
- Conclusion and Future Work

Chunk-level Detection Probability for Random Corruption

With random checking

$$\Pr(Detection) = 1 - \frac{C(N-r,n)}{C(N,n)}$$

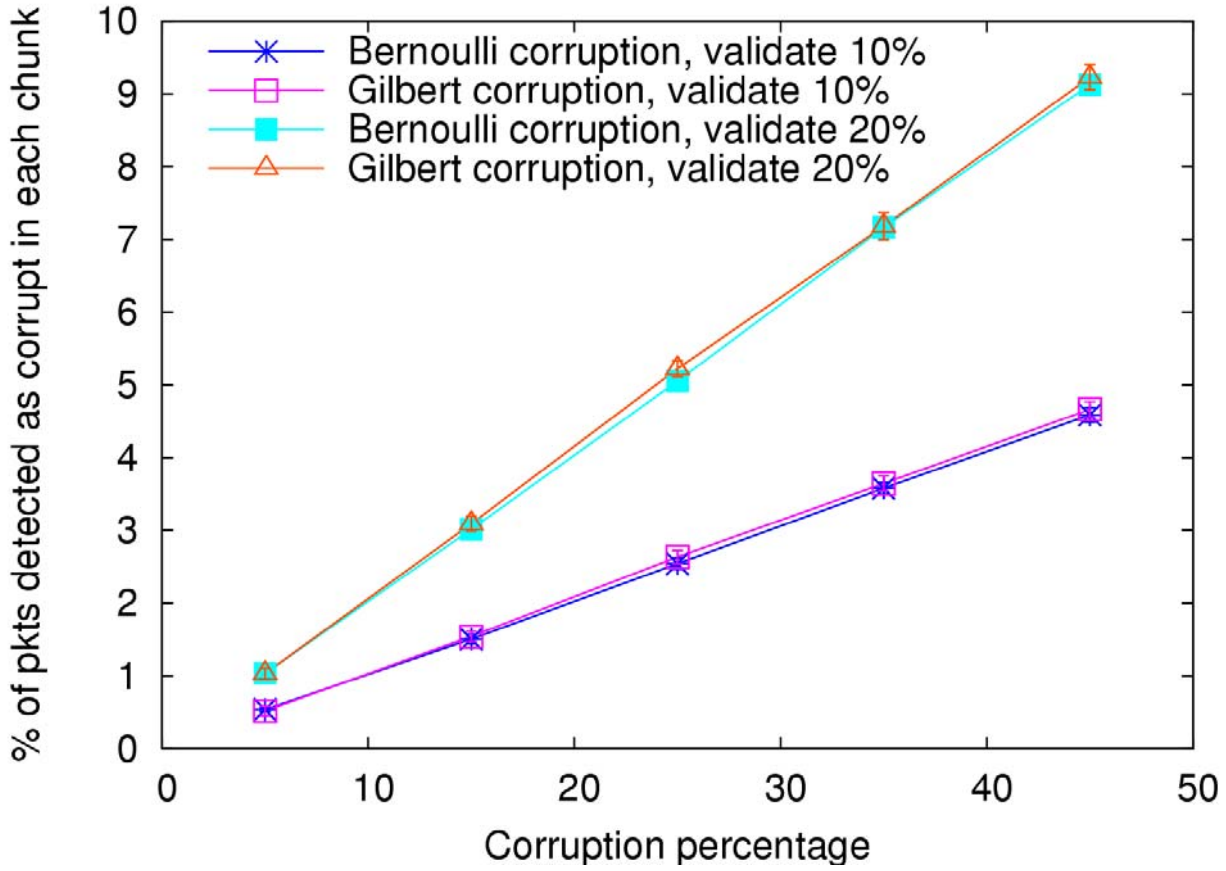
With pseudo-random checking

$$\Pr(Detection) = 1 - \frac{N}{P}\frac{C(P-r,n)}{C(P,n)}$$

N	Total number of packets in a chunk
P	Expected number of packets in a frame
n	Expected number of validated packets in a chunk (for random) or in a frame (for pseudo-random, >= 2)
r	Expected number of corrupted packets in a chunk (for random) or in a frame (for pseudo-random)
C(N, n)	Number of combinations to choose *n* out of *N* items

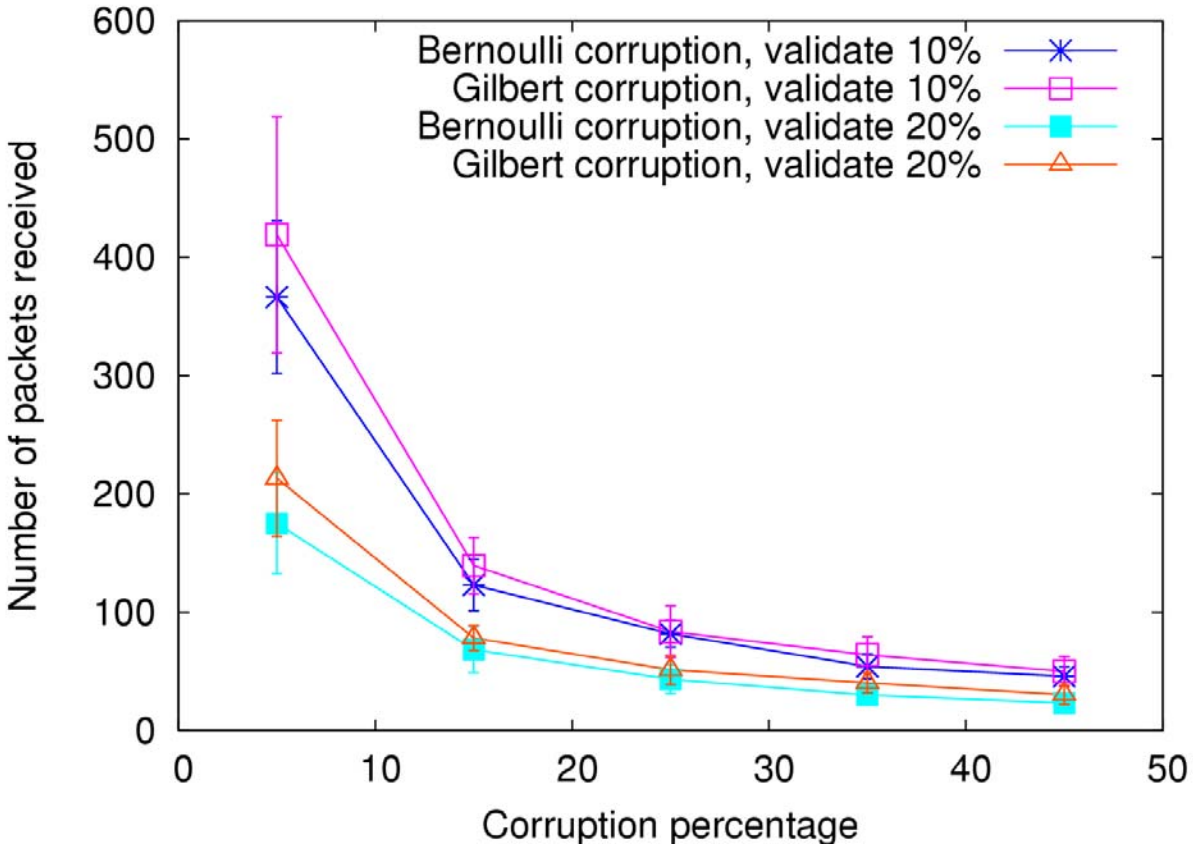
Corrupt Packets Caught

Random validation (pseudo-random similar result)



Corrupt Chunk Detection Delay – Packets Received

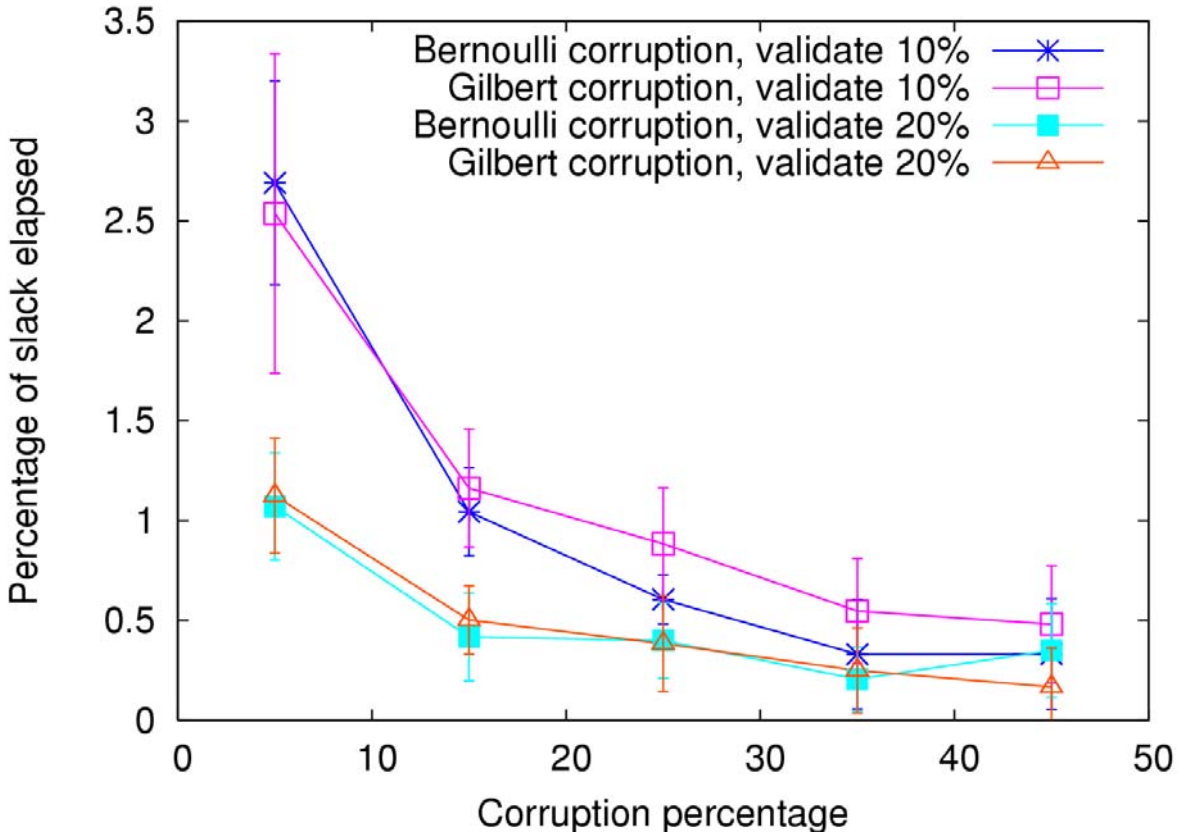
Random validation, threshold $T = 2$ (pseudo-random slightly better)



at&t

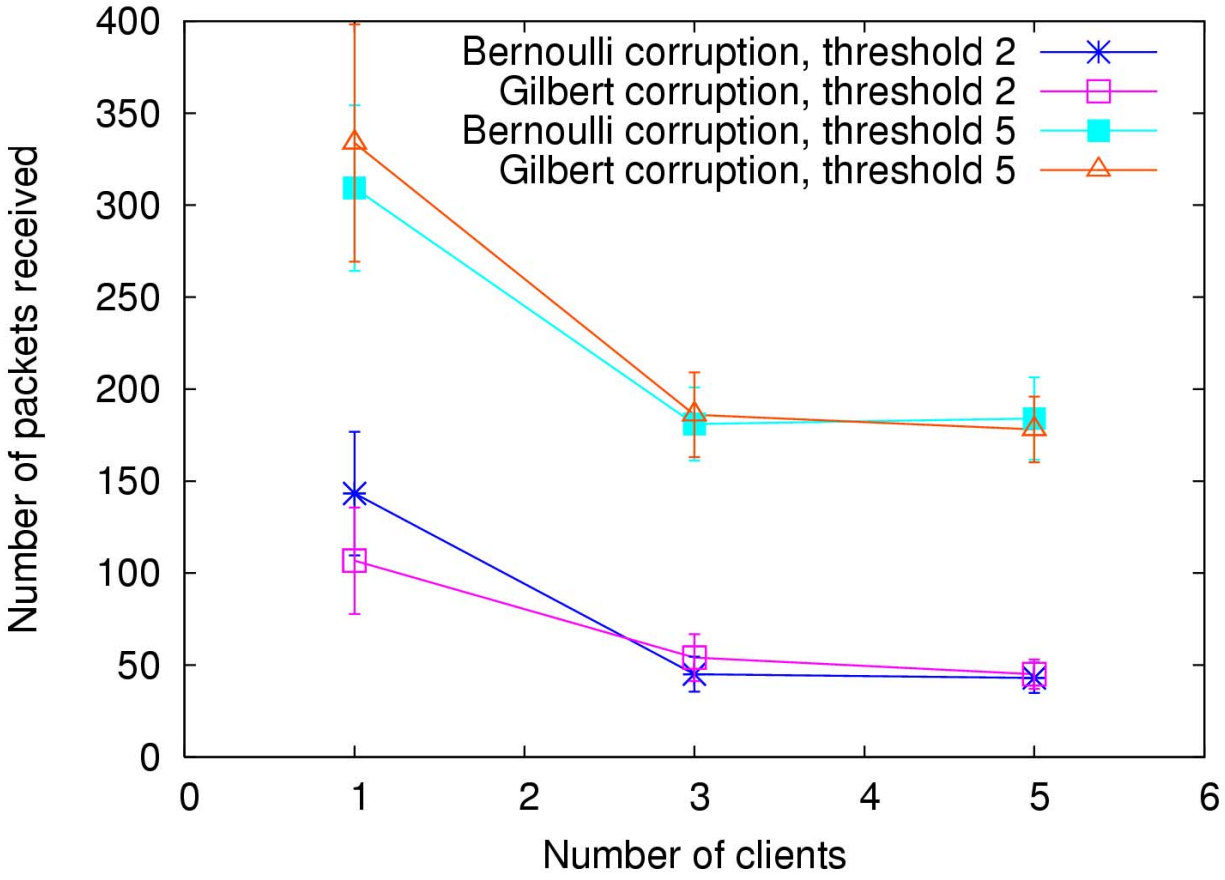
Corrupt Chunk Detection Delay – Slack Elapsed

Pseudo-random validation, threshold $T = 2$ (random slightly better)



Client Collaboration in Detection

Pseudo-random validation, n/N = 10%, r/N = 15%



Overheads

Measured on 2GHz GNU/Linux machine with 512 MB memory using MD5 hash and assuming 1472-byte data packets (1500 bytes including headers)

Computation Overhead

- 6 microseconds to compute hash of one packet, 16256 CPU cycles per hash
- 1 millisecond to hash all packets in 1 sec video

Communication Overhead

- Transfer extra 0.76 MB of hash data in “playfile” for 700 MB video when verifying 10% of packets

Storage

- Extra 16 bytes for every 1472 bytes of video

Resilience to Packet Loss

Loss probability varied from 1% to 5%.

- Packet loss did not affect our scheme
- Worst case, the average number of packets received to detect corruption increases by four more than the case without loss

Conclusion

- Proposed a packet-level (pseudo-)random sampling validation scheme for P2P video streaming
  - Loss-tolerant and real-time
  - Lightweight and practical
  - High security and detection probability
- Demonstrated and evaluated using CPM; easily extended to bit-torrent or other similar content delivery services
- Plan to adopt peer cooperation for improving corruption detection in the future

Using a trusted entity as a source of "ground truth" considerably simplifies the problem