NDNstagram - Ubiquitous Consistency (UbiCon)

Joshua Joy, Saro Meguerdichian
CS217B - Spring 2012
Outline

- Introduction
- Demo
- Solution
- Questions
Introduction

- Get the latest version of a piece of data
- IP -- only choice: ask for the latest version from the source
- NDN -- data lives everywhere
  - ask for \{version, actions\} pairs for all versions
  - maybe the latest version comes first, maybe a previous version and set of actions comes first
- NDNstagram -- distributed photo editing
Demo Time

- NDNstagram on multiple Android devices
- Take photo and apply actions on one device
- Fetch on two devices
  - Get the latest version of the data?
  - Get the original version and the list of actions, apply?
- Fetch on original device and one other
  - Original device should be much faster (apply actions)
UbiCon - UBIquitous CONsistency

- Data is ubiquitous and pervasive
- Data surrounds us in the ether in various formats and versions
- 2 dimensional consistency in time and format
- Parallely ask for all data in the ether and get back all versions of transcoded data and diffs
- By transforming the heterogeneous versioned data we arrive at the current version of data
  - $v_2 + \text{diff2} = v_3$
How to Name?

- Requesting multiple objects in parallel for various screen sizes and versions
  - Aggregate at the screen size and version
  - Route toward the producer

- Unique version names
  - Use device_id/sequence_number as the version
  - Only need sequence number locally for each image

- Infer quality from the name (e.g. large size --> small size)
  - Include size

- Determine branching structure solely from the names
  - Include previous version name
Naming Convention

- Examples
  - /ndn/ucla.edu/alice/ndnstagram/avengers/hulk/image/1280x720/alicegalaxyii/2/1280x720/alicegalaxyii/1
  - /ndn/mit.edu/bob/ndnstagram/avengers/hulk/image/960x480/bobiphone4s/3/1280x720/alicegalaxyii/2
  - /ndn/ucla.edu/alice/ndnstagram/avengers/hulk/diff/1280x720/alicegalaxyii/2/1280x720/alicegalaxyii/1
  - /ndn/mit.edu/bob/ndnstagram/avengers/hulk/diff/960x480/bobiphone4s/3/1280x720/alicegalaxyii/2

- Structure
  - Topology
  - Data Set
  - Content type
  - Image Size (transcoder, original)
  - Version (DeviceId, Sequence Number)
Meta File

- Publish meta file after each edit
  - List of image name and actions name pairs

- Each client syncs latest when fetching
  - Finds out what images and actions to fetch to get to the final version

- Branching is supported
  - Structure can be inferred from meta file
  - Client chooses which branch to fetch
  - Merging is application-specific (if needed)
Multipath

- Robust multipath fetch

- Create separate groups and dispatch
  - first group to return wins
  - cancel all other fetches
  - relies on ccnx pipelining

- Naive example
  - group1 - v3
  - group2 - v2,d2
  - group3 - v1,d1,d2
Conclusion -- Research Issues

● Naming
  ○ How to route toward specific screen sizes / versions of data while maintaining scalable naming hierarchy

● Syncing meta file

● Object request strategy
  ○ Multipath, multi-channel

● Intermediate object creation strategy

● Experiments: latency, energy, bandwidth
References

- Mark Handley. Multipath TCP: Goals and Background
- Joshua Joy; Youngtae Noh; Uichin Lee; Jihoon An; Mario Gerla. Secure Personal Content Networking over Untrusted Devices.
- Werner Vogels. Eventually consistent.
- Ellis, C.A.; Gibbs, S.J. Concurrency Control In Groupware Systems.
- Puneet Kumar & M. Satyanarayanan. Supporting Application-Specific Resolution in an Optimistically Replicated File System.
- ZHENKAI ZHU, CHAOYI BIAN, ALEXANDER AFANASYEV, VAN JACOBSON, LIXIA ZHANG. Chronos: Serverless Multi-User Chat Over NDN.