Demo Overview: HTTP Live Streaming over NetInf Transport

Bengt Ahlgren
SICS
bengta@sics.se

Arndt Jonasson
SICS
arndt@sics.se

Börje Ohlman
Ericsson
borje.ohlman@ericsson.com

ABSTRACT

We modified a commercial Android TV app to use NetInf ICN transport. It was straightforward to adapt the standard HTTP Live Streaming to NetInf naming and network service. We demonstrate that NetInf's in-network caching and request aggregation result in efficient live TV distribution.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

Keywords

Information-centric networking; NetInf; Live streaming; HLS; In-network caching

1. INTRODUCTION

Information-centric networking (ICN) [1] is an approach for designing the network of the future based on *named data objects* (NDOs) as the main abstraction. Clients request NDOs by name and publishers make NDOs available. The location of an NDO is secondary – any node holding a copy can satisfy a request for it, enabling ubiquitous in-network caching as part of the normal network service. The ICN approach was largely motivated by large scale media distribution which is dominating the Internet traffic volume, both playback of stored content and broadcast of live content.

We demonstrate live video streaming with NetInf ICN transport [2, 5] to a modified commercial TV app on Android devices. The standard HTTP Live Streaming (HLS) [6] format was adapted to the NetInf network service. The adaptation was straightforward, but required extending the NetInf implementation with support for dynamic data. The demo shows that NetInf's in-network caching and request aggregation provide efficient multicast distribution to many clients, off-loading the server and reducing network load.

ICN'14, September 24–26, 2014, Paris, France.
ACM 978-1-4503-3206-4/14/09.
http://dx.doi.org/10.1145/2660129.2660136.


```
ni://example.com/sha-256;wFNeS-K3n_2T
KRMFQ2v4iTFOSj-uwF7P_Lt98xrZ5Ro
```

Figure 1: Example 'ni' name for the named data object "Hello world!".

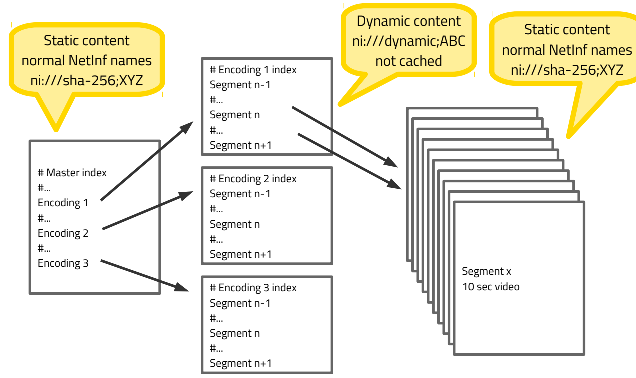

Figure 2: Live HLS video mapping to NetInf transport.

2. THE NETINF PROTOCOL

Network of Information (NetInf) [2] is an ICN architecture mainly developed in the SAIL EU FP7 project [4]. The project designed the 'ni' naming scheme for NetInf which has become IETF RFC 6920 [3]. The URI-encoded 'ni' names contain the content hash, or message digest, of the NDO, as shown in Figure 1. *Name-data integrity*, that is, verification of that the NDO received is the NDO requested, a crucial function of any information-centric network, is directly provided by the content hash.

The NetInf protocol has three major functions: `GET`, `PUBLISH` and `SEARCH`. Clients request NDOs by sending `GET` messages which are forwarded hop-by-hop by NetInf routers towards publishers. Any intermediate node that has a requested NDO, ultimately the publisher, responds with the corresponding `GET-RESP` message supplying the NDO. `PUBLISH` is used to make NDOs available, and `SEARCH` can be used to find out what NDOs are available. NetInf also has a name resolution-based model, complementing the hop-by-hop forwarding, which is not used in this demo.

3. MAPPING HLS TO NETINF

Figure 2 illustrates the three kinds of files defined by HTTP Live Streaming (HLS) and how they are mapped to NetInf. A master index file (left in the figure) lists the available video encodings, including information on bit rates. The content of the master index file is static and can therefore be named with standard hash-based

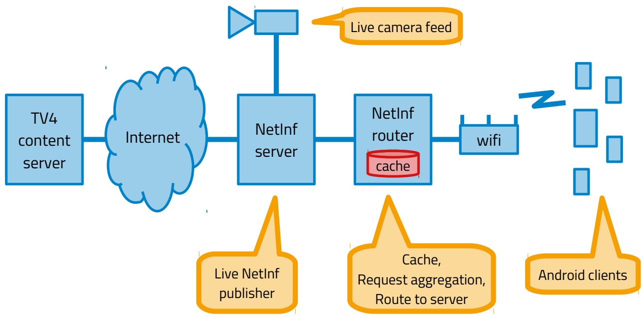

Figure 3: Setup of NetInf live streaming demo.

'ni' names. The content of the file, however, has to be modified for NetInf transport by listing the 'ni' names of the encoding-specific segment index files (middle in the figure).

When a new video segment (right in the figure) has finished recording for the live stream, the segment index files must be updated. The three most current video segments are typically listed. We have chosen to define a new name type for NDOs with dynamic content to handle this, since 'ni' names based on content hashes do not support dynamic data. The alternative would be to create new names for the segment index files after each update, but that in turn requires that the master index file has to be modified, resulting in other problems. The new name type has to use publisher signatures in order to provide name-data integrity. This is however not yet implemented in the demo. Similar to the master index file, the content of the segment index files are changed to list the video segment files by their 'ni' names.

The segment index file regularly has to be re-requested by the clients so that the clients learn the names of new video segment files. These video segment files do not change after they are recorded, and therefore regular hash-based 'ni' names work well. When a client learns the name of a new video segment file, it requests and plays back that segment.

Finally, we have turned off in-network caching for the segment index files. Otherwise the clients would never get the updates and learn the names of new video segments. It would, however, be beneficial with time-limited caching so that the distribution of these files to many clients becomes more scalable.

4. THE DEMO SETUP

Figure 3 illustrates the setup of the demo. There are two possible sources for the video: the public servers for the live and stored TV content of the Swedish TV4 channel, and our own video camera. The 'NetInf server' in the figure publishes either of these video streams in real time in the NetInf system.

A set of Android clients are illustrated at the right hand side of the figure. These run a modified version of the TV4 app that can use the NetInf network service for retrieving the HLS-encoded video streams. There is a NetInf router in between that cache NDOs, except the segment index files. The router implements aggregation of requests, which means that simultaneous requests for the same NDO from multiple clients, which is likely to happen for live streaming, result in a single request being sent upstreams to the server.

The clients and the router forward `GET` requests towards the server with default next-hop forwarding. There is a clear need for a more advanced scheme in larger setups.

An Android NetInf implementation from Ericsson is run on the clients, and the open source NetInf Python implementation developed as part of the SAIL EU project[1] is run on the server and router with Ubuntu Linux. The NetInf Python implementation has been extended with request aggregation and simple names for dynamic NDOs.

5. RESULTS AND CONCLUSIONS

We have adapted a commercial HTTP Live Streaming system to use NetInf ICN transport. The implemented system demonstrator shows the feasibility and performance of live video streaming to several clients. The caching and request aggregation of the NetInf transport result in efficient multicast to many clients. The caching removes the need for synchronisation between the clients, in contrast to the synchronous nature of IP multicast.

It was straightforward to adapt HTTP Live Streaming to use NetInf ICN transport. A naming scheme for the dynamic segment index files had to be designed and implemented for the demo.

6. ACKNOWLEDGEMENTS

This work has been supported by the EFRAIM project funded by Vinnova in the challenge-driven innovation programme, and by EIT ICT Labs. Many colleagues have considerably contributed to this work, especially Linus Sunde at Ericsson Research and Anders Lindgren at SICS. We also thank Swedish TV4, partner in EFRAIM, for making their Android app available for our experiments.

7. REFERENCES

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman. A survey of information-centric networking. *IEEE Communications Magazine*, 50(7):26–36, July 2012.

[2] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl. Network of information (NetInf) – an information-centric networking architecture. *Computer Communications*, 36(7):721–735, Apr. 2013.

[3] S. Farrell, D. Kutscher, C. Dannewitz, B. Ohlman, A. Keranen, and P. Hallam-Baker. Naming Things with Hashes. RFC 6920 (Proposed Standard), Apr. 2013.

[4] B. Kauffmann, J.-F. Peltier, et al. D.B.3 (D-3.3) final NetInf architecture. Deliverable D-3.3, version 1.1, SAIL EU FP7 Project 257448, Jan. 2013. FP7-ICT-2009-5-257448/D.B.3.

[5] D. Kutscher, S. Farrell, and E. Davies. The NetInf Protocol. Internet-Draft draft-kutscher-icnrg-netinf-proto-01, Internet Engineering Task Force, Feb. 2013. Work in progress.

[6] R. Pantos and W. May. HTTP live streaming. Internet-Draft draft-pantos-http-live-streaming-12, IETF Secretariat, Oct. 2013. Work in Progress.

[1] See `http://www.netinf.org`, and `http://sourceforge.net/projects/netinf/`.