CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Prototyping and evaluating SDN-based multicast architectures for live video streaming [Demo]
Authors	Khalid, Ahmed;Zahran, Ahmed H.;Sreenan, Cormac J.
Publication date	2017-10
Original Citation	Khalid, A., Zahran, A. H. and Sreenan, C. J. (2017) Prototyping and Evaluating SDN-based Multicast Architectures for Live Video Streaming [Demo], 42nd IEEE Conference on Local Computer Networks (LCN) Singapore, 9-12 October.
Type of publication	Conference item
Link to publisher's version	https://www.ieeelcn.org/prior/LCN42/Program_demos.html, http://hdl.handle.net/10468/5308
Rights	© 2017 the authors
Download date	2026-10-05 12:29:15
Item downloaded from	https://hdl.handle.net/10468/5309

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Prototyping and Evaluating SDN-based Multicast Architectures for Live Video Streaming

Ahmed Khalid
University College Cork
Cork, Ireland
Email: a.khalid@cs.ucc.ie

Ahmed H. Zahran
University College Cork
Cork, Ireland
Email: a.zahran@cs.ucc.ie

Cormac J. Sreenan
University College Cork
Cork, Ireland
Email: cjs@cs.ucc.ie

***Abstract*—Software Defined Networking (SDN) is an emerging approach for network programmability, with the capacity to initialize, control, change, and manage network behavior dynamically via open interfaces. The rise of SDN presents an opportunity to overcome the limitations of rigid and static traditional Internet architecture and provide services like inter-domain network layer multicast for live video streaming. In this demonstration we present a platform to evaluate and compare SDN-based multicast architectures for live streaming and benchmark their performance against standard IP unicast. The platform is equipped with two graphical user interfaces (GUI). A Panoramic UI provides a mechanism to modify various evaluation parameters and monitor the effect on output in form of graphs and live statistics. An Animator UI displays traffic flows over the chosen network topology and offers packet level information for the ongoing video streams. We also present a prototype of mCast implemented on our platform. mCast is a novel SDN-based multicast architecture for live video streaming over the Internet. The feedback from our platform and its GUIs showcases how mCast can save network and system resources while improving the video quality for clients.**

I. Introduction

Software Defined Networking (SDN) is a network paradigm that separates the control and forwarding planes, enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services [1]. A logically centralized controller, with a global view of network, can monitor traffic flows, make forwarding decisions and install efficient rules at runtime. This flexibility and control can be exploited to provide services like inter-domain network layer multicast. As more than 40% of video traffic over Internet is live [2], significant amount of system and network resources can be saved by using multicast. Over the past few years some SDN-based multicast frameworks [3], [4], [5] have been proposed to offer content and/or network providers with sufficient control to realize a video streaming service.

In this demonstration we present a modular platform to evaluate and compare SDN-based multicast architectures and algorithms for live video streaming and benchmark their performance against standard IP unicast. We stream actual video content in High Definition (HD) quality over an emulated network and measure various network and application metrics. The platform consists of two graphical user interfaces (GUI). A Panoramic UI allows modification of various network and video parameters (Table 1) to vary the evaluation scenario and monitor the effect on output in form of graphs and live statistics. An Animator UI displays the chosen network topology and shows traffic flows from video servers to the clients in real-time. The UI helps in visually identifying patterns in communication and better understand the working of the underlying protocols and algorithms.

TABLE I
UI Parameters to Modify Network and Video Settings

Network	
Clients	Total number of clients in the ISP network that will request a video stream
Network Topology	Options include: Star, Mesh and Tree topology
Link capacity	The maximum link bandwidth for internal links of the ISP network
Video	
Video Servers	Number of CDN servers with video content to stream
Duration	Total duration (in minutes) for which the video will be streamed.
Quality	To choose High Definition (HD) or Standard (SD) video quality
Decoder	Options include: RAW - Do not display video at clients and H.264 - Displays video at the clients as they receive it

We also implement a prototype of mCast [3], a novel SDN-based multicast architecture for streaming live video content from servers in Content Delivery Networks (CDN) to clients in Internet Service Providers (ISP). This prototype illustrates all the elements and modules required to implement and evaluate SDN-based multicast architectures on our platform. Users of the platform can replace any or all modules with their own, to evaluate and compare the performance of their proposed design.

The rest of the paper is structured as follows. In Section 2 we provide a brief overview of mCast and its components. In Section 3, we present our evaluation platform. In Section 4 we give an overview of the demonstration and in Section 5 we provide a list of the equipment required for the demonstration setup.

II. mCast Overview

mCast presents a novel SDN-based solution that exploits network layer multicast to reduce the resource utilization

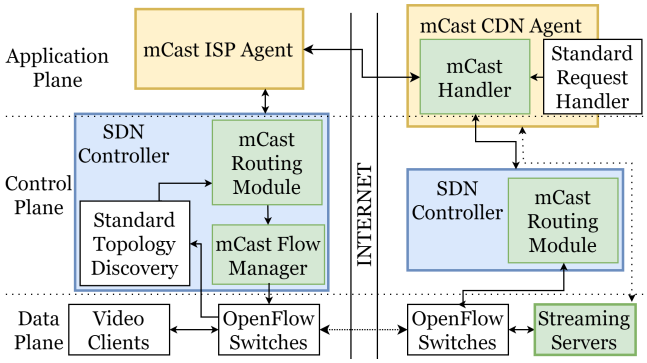

Fig. 1. Architecture of mCast.

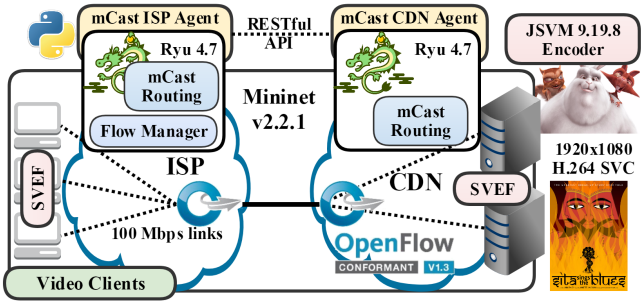

Fig. 2. Testbed Setup

in ISPs and CDNs while providing CDNs with full control over their clients. Unlike IP multicast, mCast provides CDNs with features like group management, authorization, billing policies, data privacy and security. The clients do not need to be modified because mCast converts the stream back to IP unicast at the last hop, making the delivery transparent.

Figure 1 illustrates key mCast architectural elements. mCast employs agents in both ISP and CDN. These agents are responsible for communication with each other as well as the control plane of SDN. A CDN can choose to switch from IP unicast to mCast when doing so will reduce the overall cost. Once the decision has been made, mCast CDN Agent provides mCast ISP Agent with a list of its clients that are watching the stream and can be served with multicast. The mCast ISP Routing Module then uses this information to construct a dynamic multicast tree and compute forwarding paths.

Based on the constructed tree, mCast Flow Manager installs multicast entries in network nodes with higher priority than IP unicast, ensuring that clients are served with mCast whenever possible. In addition, mCast Flow Manager installs transparency rules on the egress switch. Before forwarding a packet to the client, this rule modifies the destination IP and port to match the IP and Port address of the client. Hence the client receives the packet just as it would in IP unicast.

For CDNs and ISPs, keeping their network infrastructure private is of utmost importance and they tend not to reveal information such as network topology, available bandwidth, or routing paths. mCast is designed in a way that no such information needs to be disclosed. Both ISPs and CDNs can manage their clients and network in their own way and just share the identity of clients to be served with mCast. This resolves any concern that a CDN might have in terms of user or data privacy.

To show the feasibility and robustness of mCast, We implemented mCast on the platform described in Section 3 and benchmarked the performance against standard IP unicast. The results showed that mCast can decrease link utilization by more than 50% and completely eliminate packet losses by avoiding any congestion in the network. The quality of video received by users was also improved as no data was lost and network delays were lower.

III. Platform Overview

Our platform for performance evaluation is based on an emulated testbed. Figure 2 illustrates our testbed setup. We use Mininet[1] for emulation which is well-known and widely used, and provides a means of creating a virtual network topology on a single computer. We run Mininet on a laptop with Ubuntu 16.04. We implement two separate SDN-based domains, for ISP and CDN, connected over a high speed link. For the CDN, we use a tree topology with a depth and fan-out equal to one. For the ISP domain, star, mesh or tree topology can be chosen to see the behavior of the architecture with different underlying network topologies.

Each domain is managed by a separate python-based Ryu controller[2] over which control and application planes are implemented. We use OpenFlow 1.3 protocol for the SDN southbound interface. mCast CDN Agent runs as a separate module over the CDN controller. mCast ISP Agent runs as a web server using a RESTful API. CDN can communicate with ISP using HTTP requests that are received by the RESTful API in mCast ISP Agent and passed to the SDN controller of ISP using Web Server Gateway Interface (WSGI).

We use two open source videos "Big Buck Bunny" (*bbb*) and "Sita Sings the Blue" (*sstb*) to be streamed from the CDN. The raw videos were encoded using JSVM framework [6]. Nine minutes of each video were encoded at 1920x1080 HD resolution with a Group of Picture (GOP) size of 8 at a frame rate of 25 fps, yielding a bitrate of 2 Mbps.

We used C++ implementation of video client and server from the scalable video evaluation framework (SVEF) [7]. However, we modified the server implementation to act as a live streaming server that is capable of dynamically streaming the content to more than one client as it receives content requests. Additionally, we implemented an API for the interaction between content servers and the mCast CDN agent.

IV. Demonstration Overview

This demonstration offers a means of modifying various network and video settings (Table 1), and monitoring the effects of these changes on the network and system metrics in real-time. Users can modify these settings through a Panoramic UI (Figure 3). The UI also offers a means of viewing performance

[1]http://mininet.org/
[2]https://osrg.github.io/ryu/

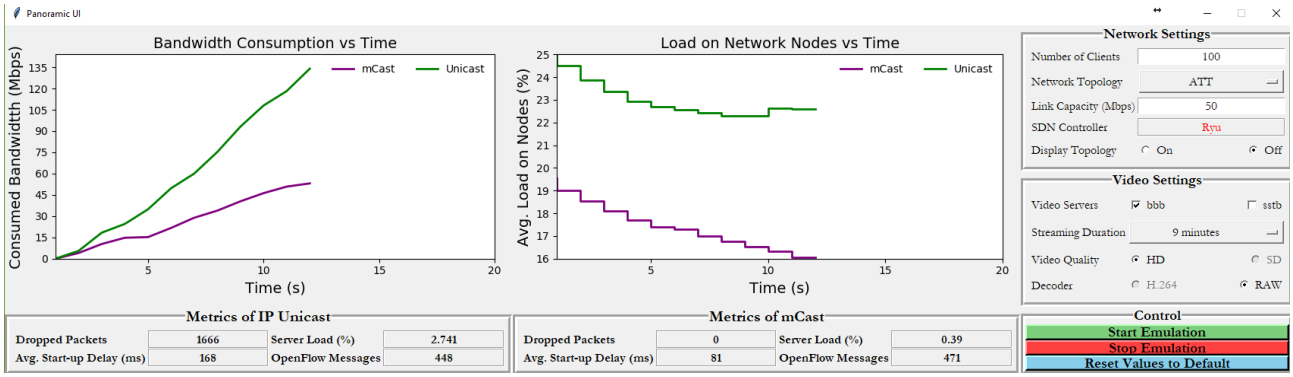

Fig. 3. Panoramic UI utilized to modify network and video settings and view performance metrics in real-time

metrics mandated by the input settings. These metrics are displayed in the form of graphs and statistics. The information shown includes bandwidth consumption in the ISP network plotted over time, average load on network nodes including switches and controllers plotted over time, number of dropped packets, average load on content servers in CDN, average start-up delays for the clients and total number of OpenFlow messages in the ISP network. For interactive demonstration, conference participants will be able to see the performance of mCast against IP unicast with either default parameters or custom parameters of their choice using Panoramic UI.

In addition, a second UI (Figure 4) displays the chosen network topology including clients, servers and the SDN controllers. The traffic flows can be viewed in real-time as the packets travel from servers to the clients. We use MiniNAM [8] for this purpose. MiniNAM is a utility that provides real-time animation of networks created by the Mininet emulator. A user can view selective flows with options to color code packets based on packet type and source node. For simultaneous comparison of mCast and IP unicast, we create an instance of MiniNAM for each approach in a separate virtual machine. With the packet level information provided by MiniNAM, users can visually identify patterns in communication and better understand the working of mCast.

V. Required Equipment

To demonstrate our platform we will use a single laptop with virtual machines running on it. Setup of the demonstration requires five minutes. The exact requirements in terms of equipment are provided in the list below:

- A monitor (32” or larger) to increase the visibility of the GUIs
- A table that can support a laptop and the monitor
- Power supply for a laptop and the monitor
- An HDMI cable
- Poster wall
- Internet access is *not* required

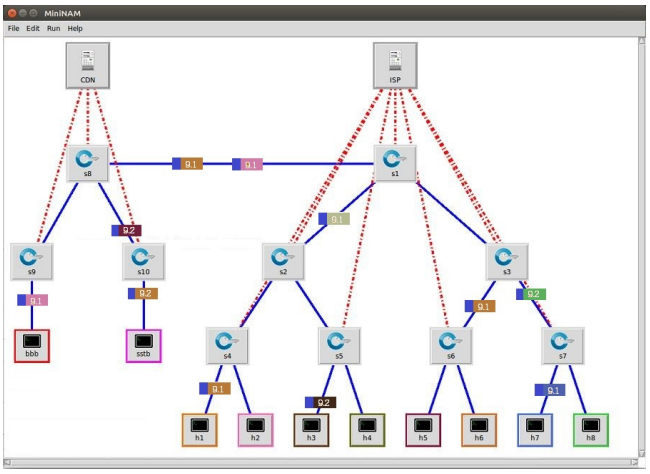

Fig. 4. A snapshot of MiniNAM with Tree topology

Acknowledgment

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number: 13/IA/1892.

References

[1] E. Haleplidis, K. Pentikousis, S. Denazis, J. Salim, D. Meyer, and O. Koufopavlou, *Software-Defined Networking (SDN): Layers and Architecture Terminology*, January 2015, https://www.rfc-editor.org/rfc/rfc7426.txt.

[2] “Cisco Global Cloud Index: Forecast and Methodology, 2015-2020 (White Paper),” Cisco Public, Tech. Rep., 2016.

[3] A. Khalid, A. H. Zahran, and C. J. Sreenan, “mCast: An SDN-based Resource-Efficient Live Video Streaming Architecture with ISP-CDN Collaboration,” *Proceedings of the IEEE Conference on Local Computer Networks (LCN), (to appear)*, October 2017.

[4] J. Ruckert, J. Blendin, and D. Hausheer, “Software-Defined Multicast for Over-the-Top and Overlay-based Live Streaming in ISP Networks,” *Journal of Network and Systems Management*, vol. 23, no. 2, pp. 280–308, April 2015.

[5] K. Noghani and M. Sunay, “Streaming Multicast Video over Software-Defined Networks, journal = Proc. of the 2014 IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems (MASS),” pp. 551–556, October 2014.

[6] J. Reichel, H. Schwarz, and M. Wien, “Joint scalable video model JSVM-9,” *Joint Video Team*, January 2007.

[7] A. Detti, G. Bianchi, C. Pisa, F. Proto, P. Loreti, W. Kellerer, S. Thakolsri, and J. Widmer, “SVEF: an open-source experimental evaluation framework for H.264 scalable video streaming,” *IEEE Symposium on Computers and Communications*, July 2009.

[8] A. Khalid, J. J. Quinlan, and C. J. Sreenan, “MiniNAM: A network animator for visualizing real-time packet flows in Mininet,” *20th Conference on Innovations in Clouds, Internet and Networks*, March 2017.