FlexStream: Towards Flexible Adaptive Video Streaming on End Devices using Extreme SDN

Ibrahim Ben Mustafa

Old Dominion University

iben@cs.odu.edu

Tamer Nadeem

Virginia Commonwealth University

tnadeem@vcu.edu

Emir Halepovic

AT&T Labs - Research

emir@research.att.com

ACM Multimedia Conference, Seoul, Korea, 22-Oct-2018

All presented information does neither reflect nor imply an actual business case for AT&T and associated companies.
They are generalized data used to assess performance of the algorithms only.

Mobile Video Traffic

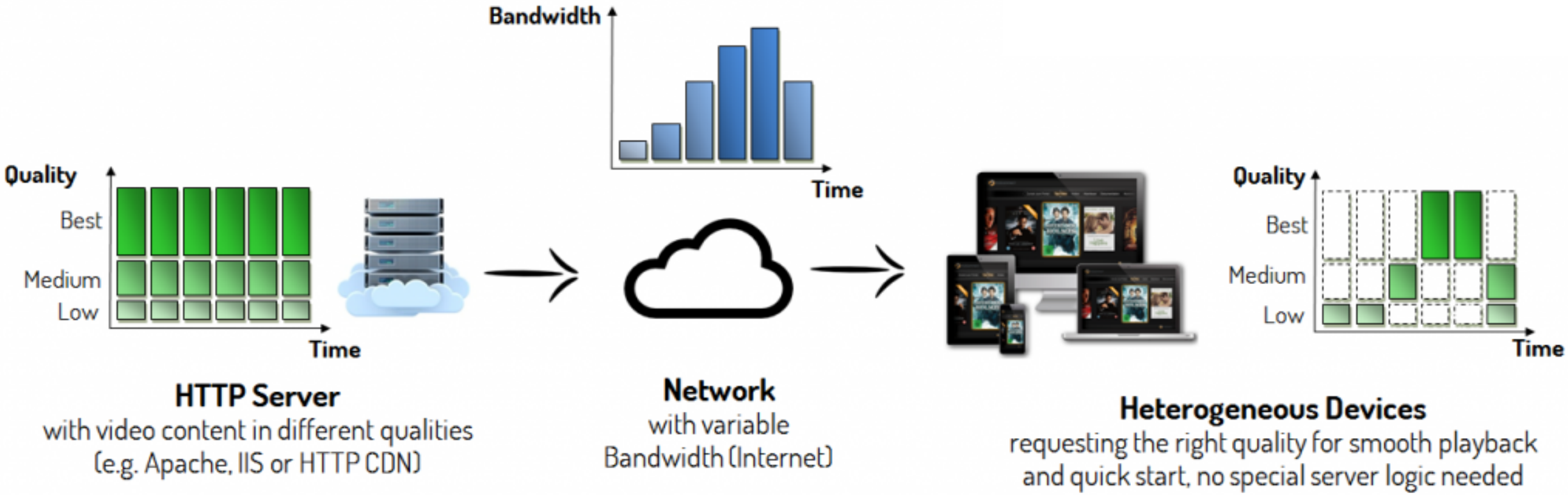
- Witnessing tremendous growth in mobile data traffic.
- Mobile data traffic is predicted to increase seven fold between 2016 and 2021
  - Mobile Video would be responsible for more than %75 by 2020.(*)

Live Streaming

Live broadcasting

(*) Cisco Visual Networking Index - 2017

HTTP Adaptive Streaming (HAS)



Images retrieved from: https://bitmovin.com/

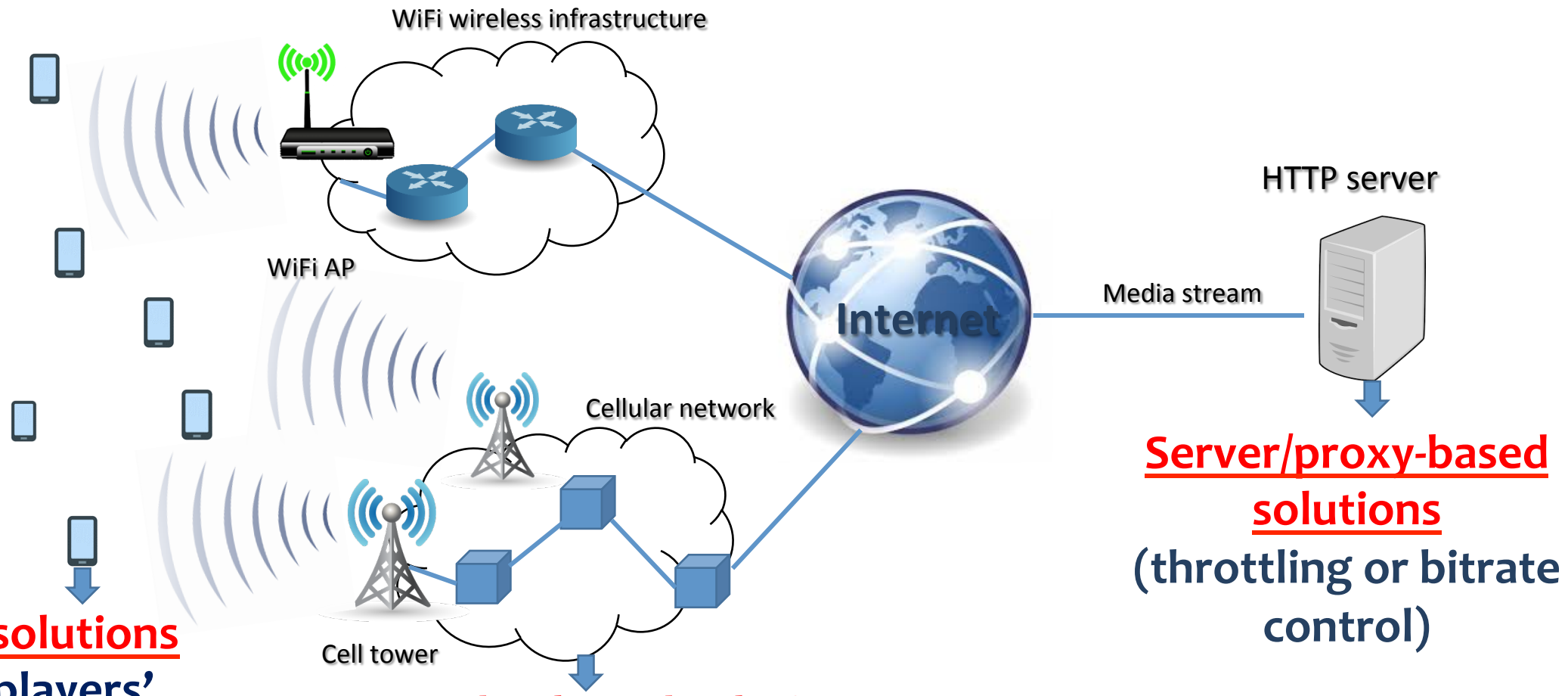
Problem & Related Work

Performance Issues with HAS

- When HAS players compete over the bottleneck:
  - Instability in the quality
  - Playback stalls
  - Unfairness
  - Long startup delay
- Root cause: ON/OFF traffic pattern(*)

(*) Saamer Akhshabi, Lakshmi Anantakrishnan, Ali C Begen, and Constantine Dovrolis. 2012. What happens when HTTP adaptive streaming players compete for bandwidth? In ACM NOSSDAV, June 2012.

Existing Solutions



Client-based solutions
(improving players' adaptation algorithm)

Edge-based solutions
(traffic management and control)

Server/proxy-based solutions
(throttling or bitrate control)

Issues with existing Solutions

- Existing solutions are either:
  1. Not effective, since they can not:
     - Address the main performance issues.
     - Comply with network policies.
  2. Invasive: Players have to follow specific adaptation logic.
  3. Not generic: Specific for HAS.
  4. Costly: Require large and special-purpose network infrastructure.
  5. Infeasible (in practice):
     - Requires CDN edge server changes.
     - Require player feedback and interactions.

Our Solution: FlexStream

- SDN-based framework that leverages:
  - **Centralized/edge component:**
    - Enables global view of network condition.
    - Context-aware through end device feedback.
    - Specifies a policy controlling resource allocation, using an optimization function.
  - **Distributed SDN component:**
    - Monitors and reports various context information.
    - Implements network policies.
    - Offloads fine-grained functionality to the end device.

FlexStream Benefits

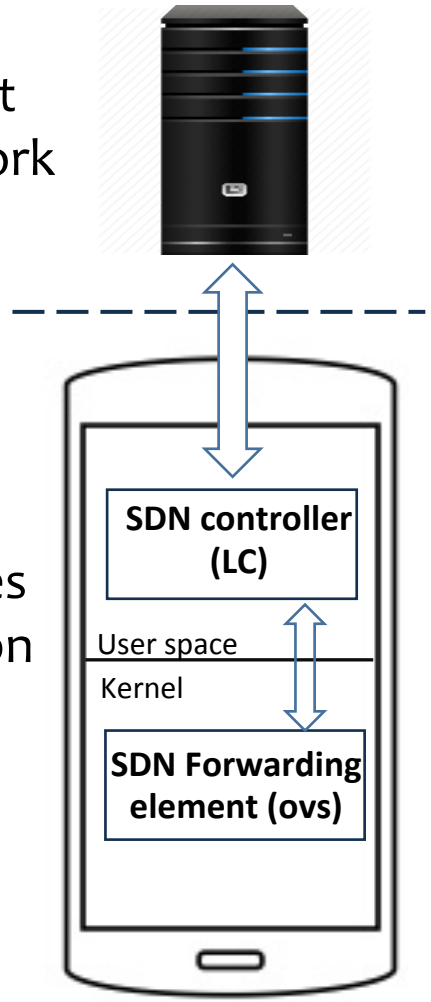
- Offloads intrusive or resource-demanding tasks from the network to end devices.
- Allows for fine-grained and intelligent management of bandwidth based on real time context awareness and specified policy.
- Flexible implementation of network policies.
- Improves video QoE:
  - Reduces quality switching by 81%, stalls by 92%, and startup delay by 44%.
- Offers universal approach to work across network technologies, WiFi and cellular.
- Has no dependency on the internal network support.

System Overview and Architecture

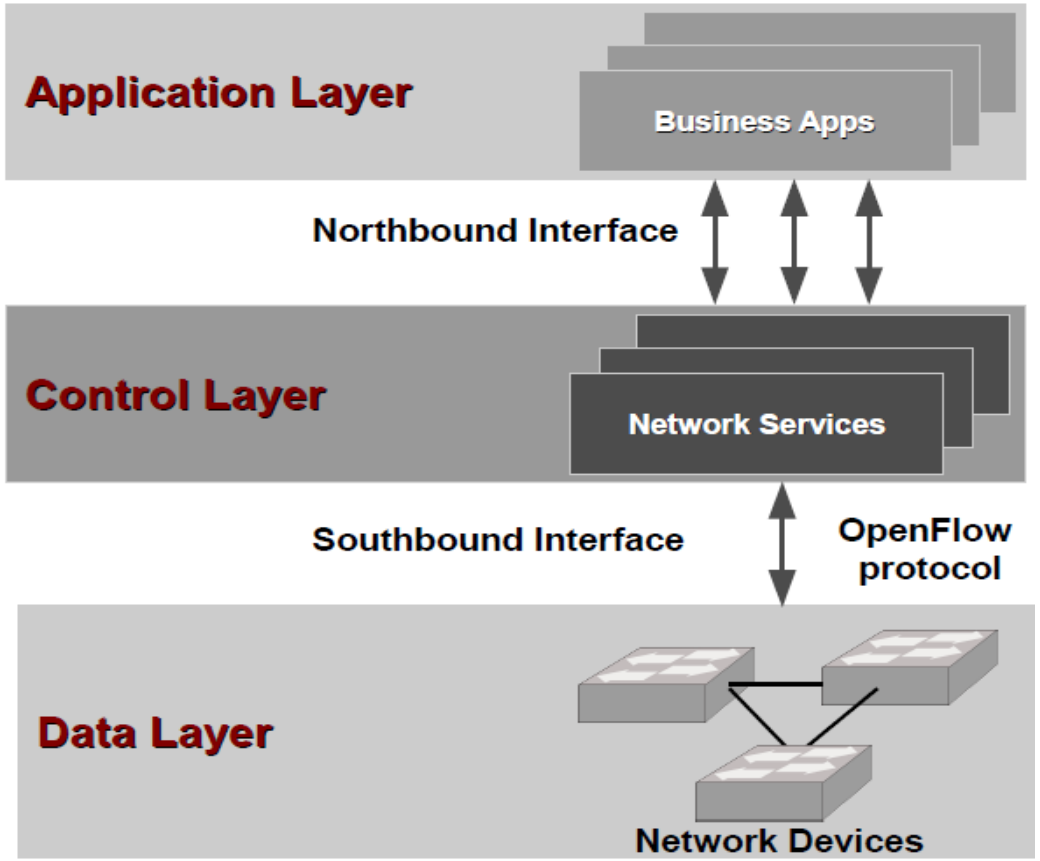
Utilizing SDN on End Device (*extreme SDN*)

Resource management (cloud or wireless network infrastructure)

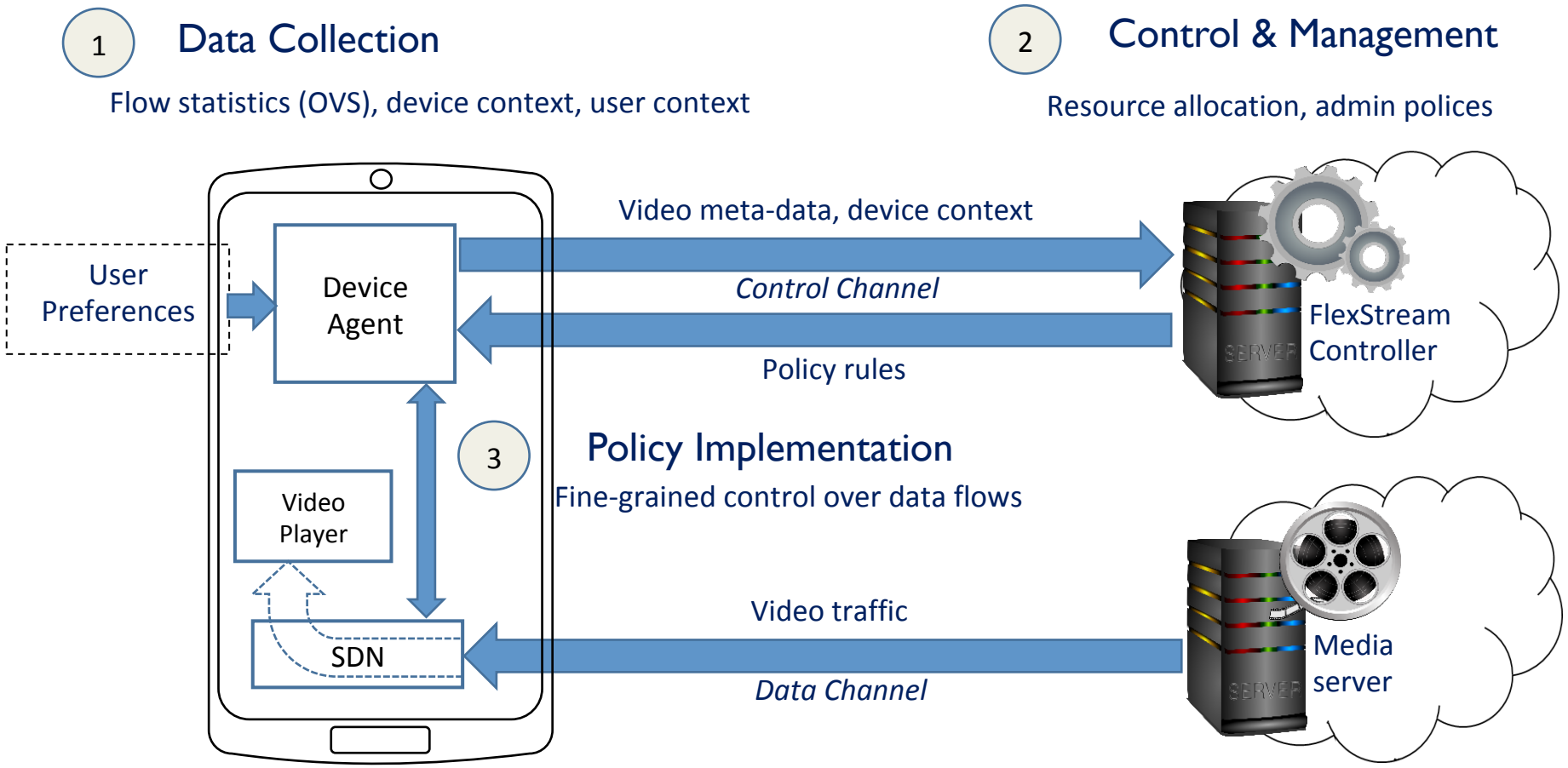
Applying network policies using SDN components on an end device.



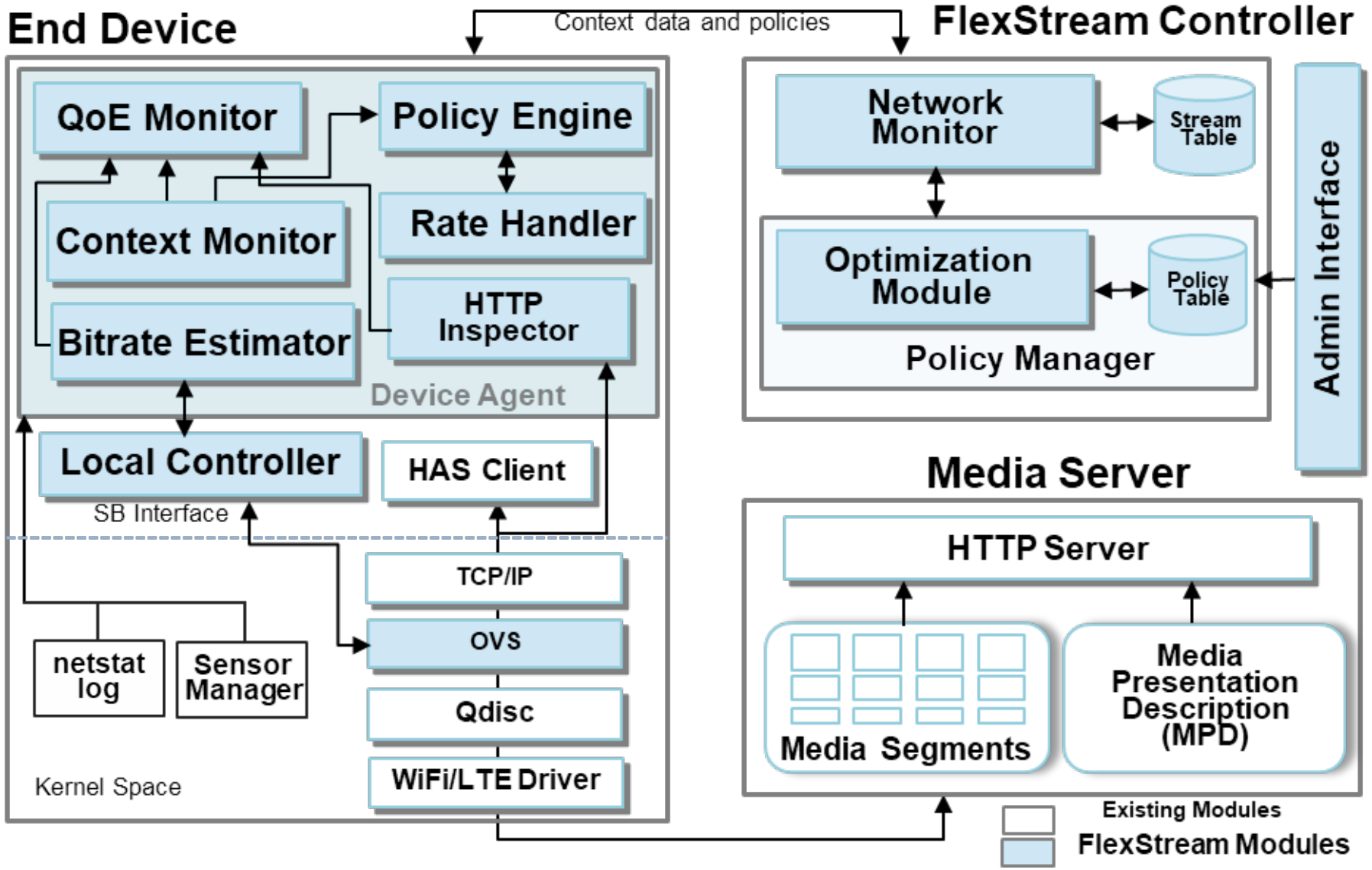
SDN Planes and Layers



FlexStream – Overview

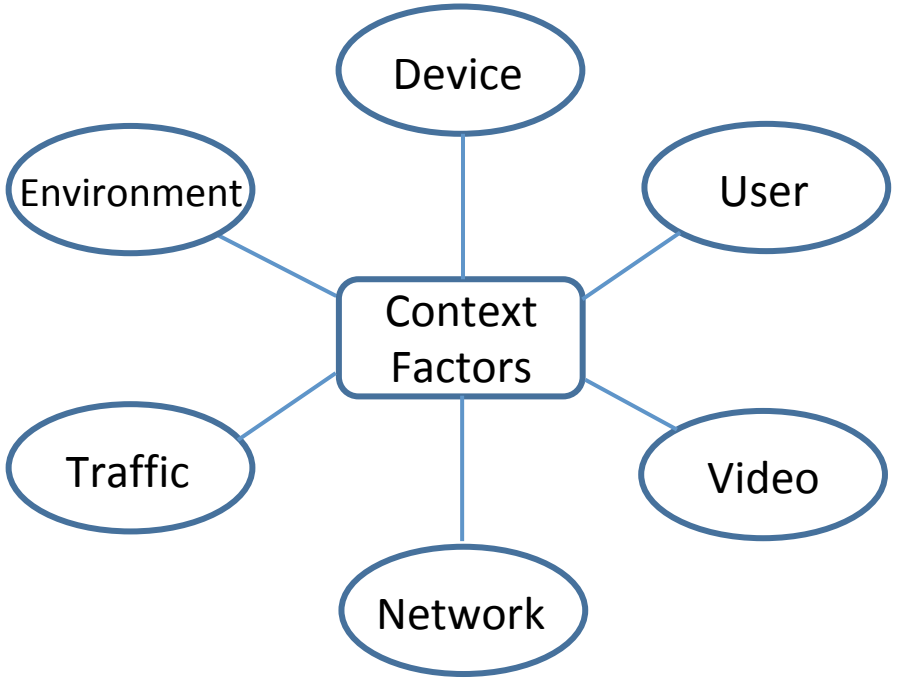


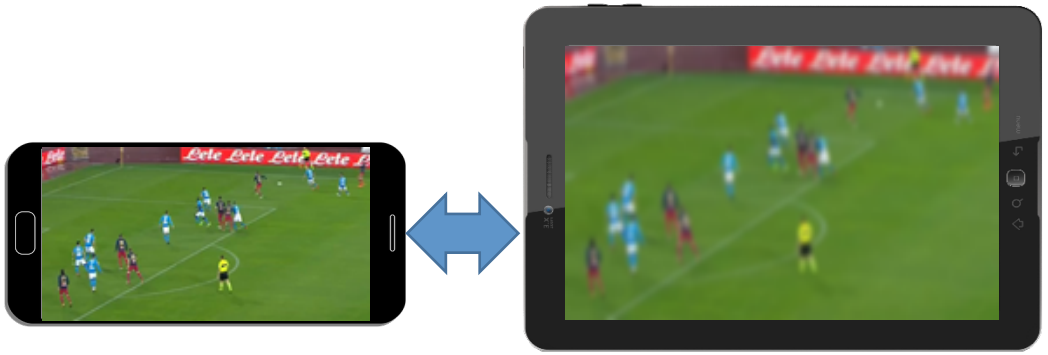
FlexStream – Architecture



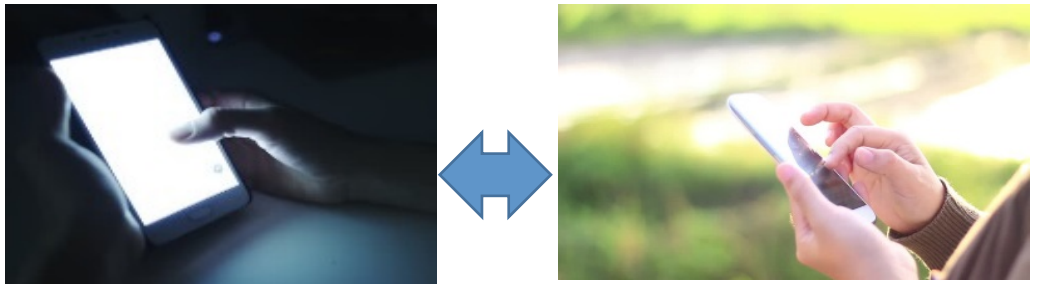
FlexStream – Context-Awareness

- Supports various management policies based on the different contexts for:
  - Fair and balanced watching experience.
  - Maximizing videos bitrates.
  - Better bandwidth utilization.





Screen Size



Surrounding Luminance

FlexStream Controller – Optimization Module

Optimization Problem

$$\max_{x_{ij}} \sum_{i=1}^{N} \sum_{j=1}^{K_i} (u_{ij} - \mu\delta_{ij})x_{ij}$$

$$\text{subject to} \sum_{i=1}^{N} \sum_{j=1}^{K_i} (\epsilon r_{ij})x_{ij} \leq B$$

$$\sum_{j=1}^{K_i} x_{ij} = 1,\ x_{ij} \in 0, 1\ \forall i$$

Utility Function

$$u_{ij} = \prod_{l=1}^{a} \beta_{il} . \log(r_{ij})$$

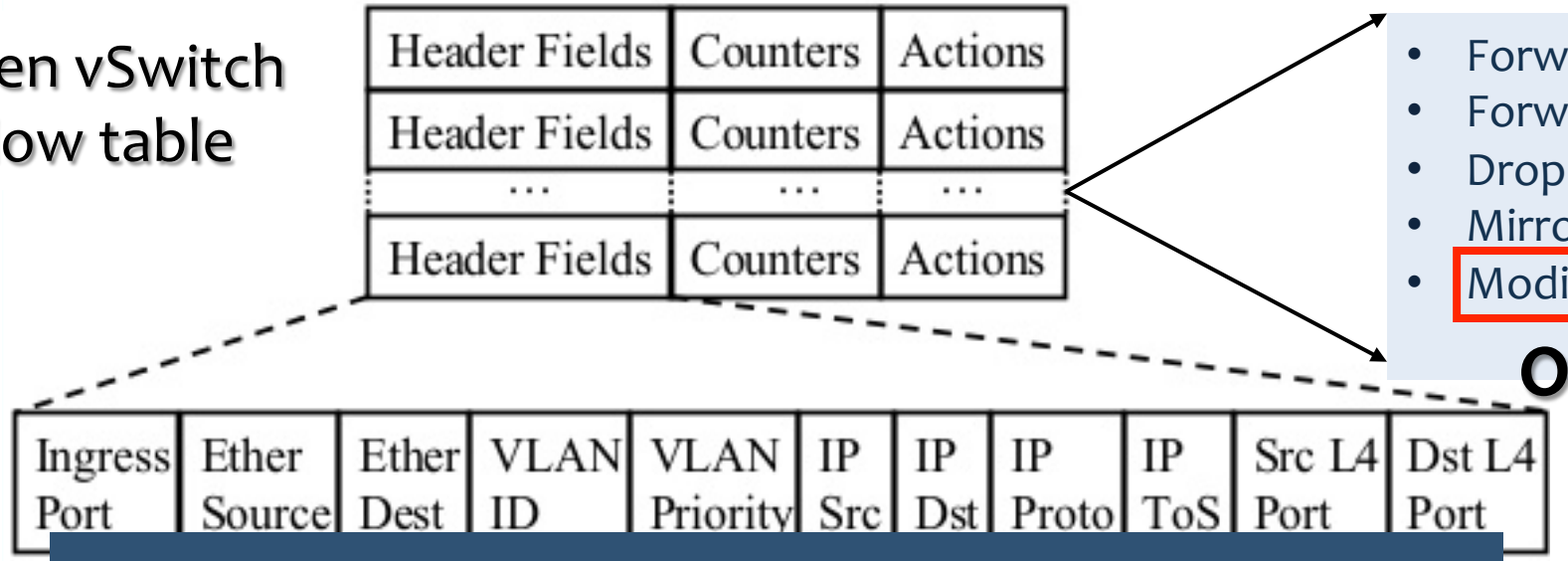
Penalty Function

$$\delta_{ij} = \begin{cases} |r_{ij} - r_{ic}|s_i + (m - \lceil \frac{t_i}{k} \rceil), & t < t_{thresh} \\ |r_{ij} - r_{ic}|s_i, & t \geq t_{thresh} \end{cases}$$

Implementation Challenges

- Extending SDN planes to enable controlling the data rate on the end device.

Open vSwitch flow table



- Forward the packet to an output port
- Forward to SDN controller
- Drop the packet
- Mirror the packet
- Modify packet headers.

OVS does not support data rate control

OVS & Open Flow protocol are extended to control and limit data rate through TCP flow control mechanism

Implementation Challenges

- It is possible to bind OVS to Wi-Fi interface, but not to the cellular interface:
  - Uses different technologies and protocols to connect to its base station.
  - Moving the IP address of the cellular interface to OVS immediately breaks the connection with the base station.

- Typical solution?
  - Using a Wi-Fi access point as a mediator, but does not allow for direct experimentation.

- FlexStream?
  - Installing a number of rules to the OVS flow table to rewrite the source/destination IP and MAC addresses with OVS addresses to force all traffic to pass through OVS.

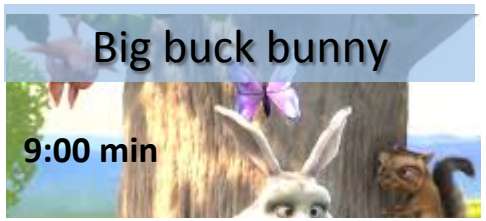
Evaluation

Evaluation

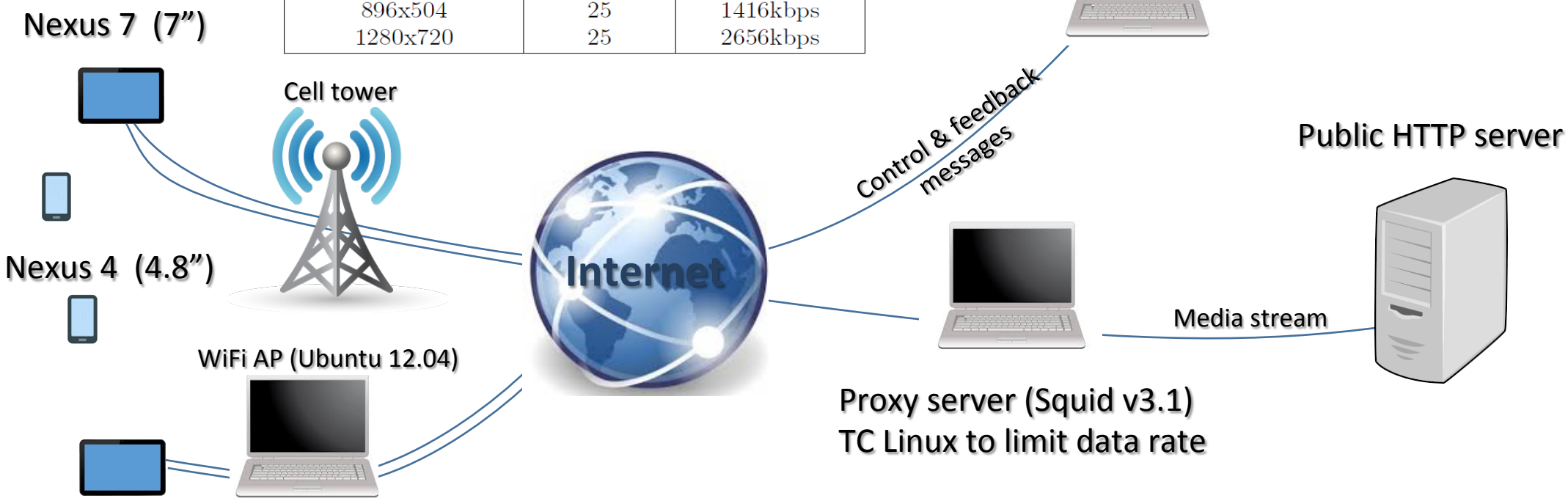
- **Quality Metrics**: Stability, fairness, stalls, and startup latency.
- **Scenarios:** Static Bandwidth and Dynamic Bandwidth
- **Experiments**
  - Basic: 3 real players in a real network.
  - Extended: 12 emulated players & server, real network.
- **Context:** User priority, screen size, link condition, background traffic, and surrounding luminance.
- **Overheads:** Computation and bandwidth.

Setup for Basic Experiments

OVS (v1.9)
OpenFlow (v1.2)
OVS-VSCTL(v1.9)
OVS-OFCTL(v1.9)
GPAC(v0.6.2-DEV)
Device Agent



Encoded Frame size	Frame Rate	Approx Bitrate
512x288	25	449kbps
704x396	25	843kbps
896x504	25	1416kbps
1280x720	25	2656kbps



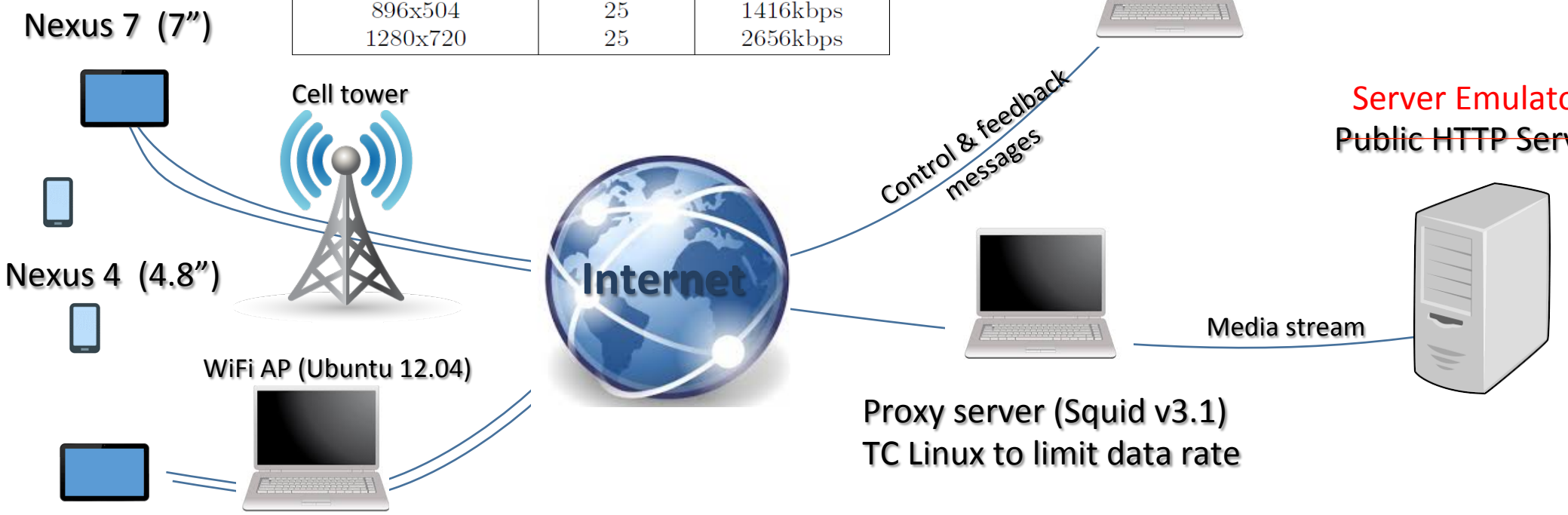
Setup for Extended Experiments

Dummy video segments equivalent in size and distribution to those used in the basic experiment

Encoded Frame size	Frame Rate	Approx Bitrate
512x288	25	449kbps
704x396	25	843kbps
896x504	25	1416kbps
1280x720	25	2656kbps

OVS (v1.9)
OpenFlow (v1.2)
OVS-VSCTL(v1.9)
OVS-OFCTL(v1.9)
~~GPAC(v0.6.2-DEV)~~
Device Agent

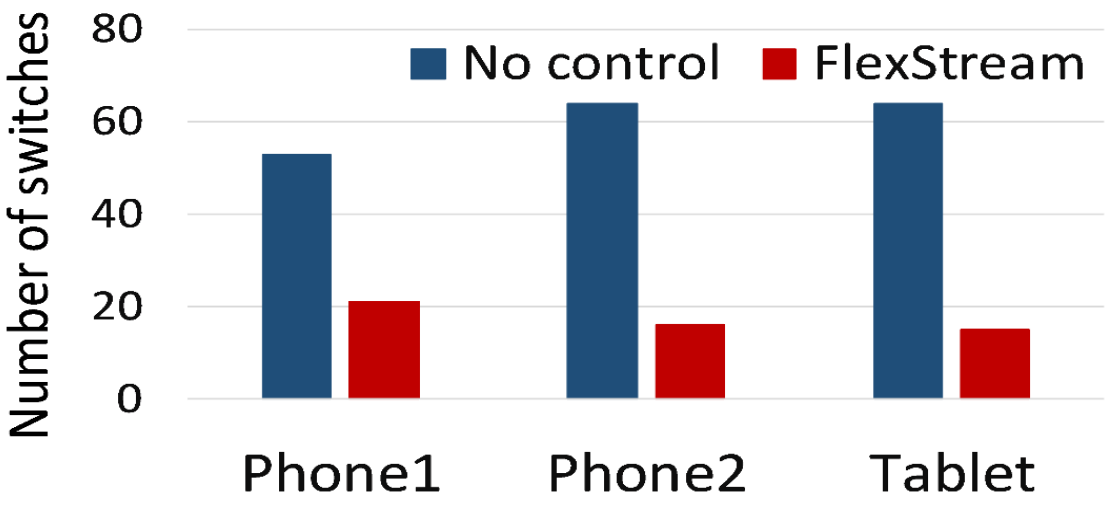
Player emulator



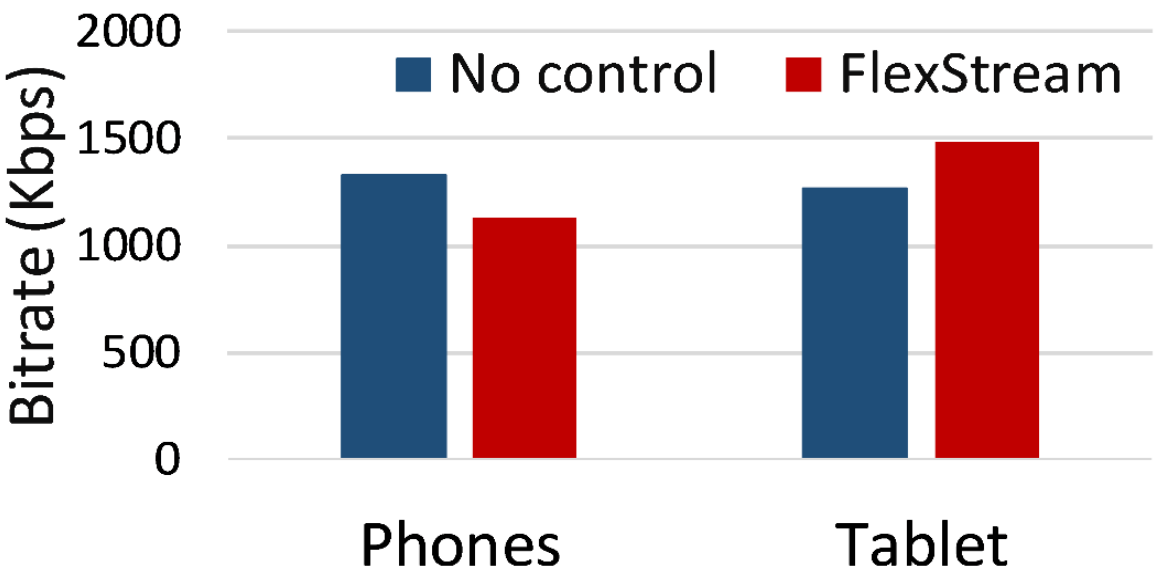
Basic Experiments

Experiments with different network capacities, starting from 2500 Kbps to 8500 Kbps with an increase of 1500 Kbps.

Average bitrate switches per device

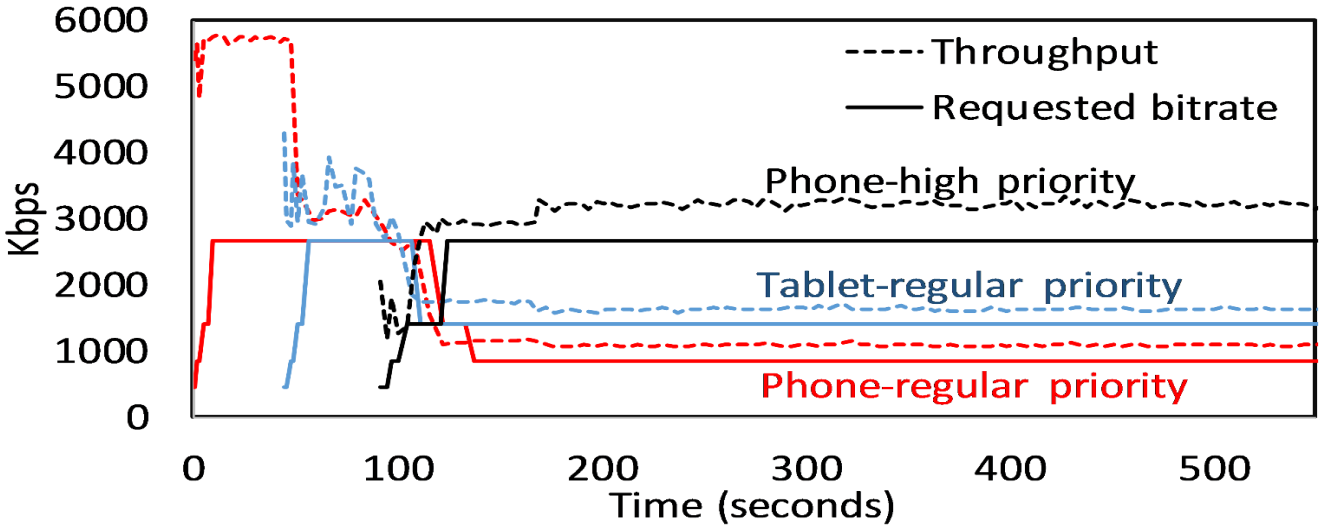


Balanced QoE for phones and tablet

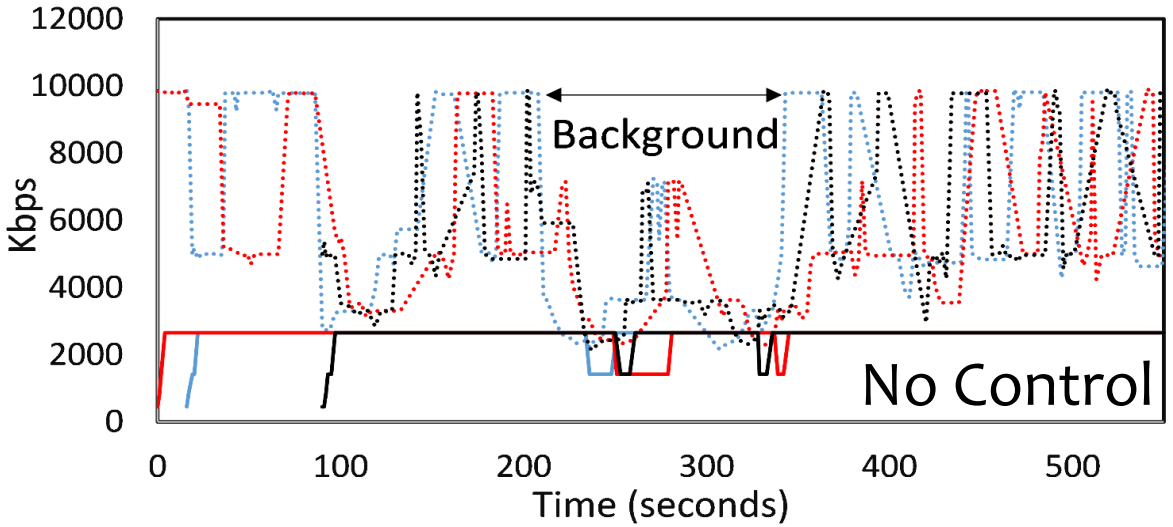


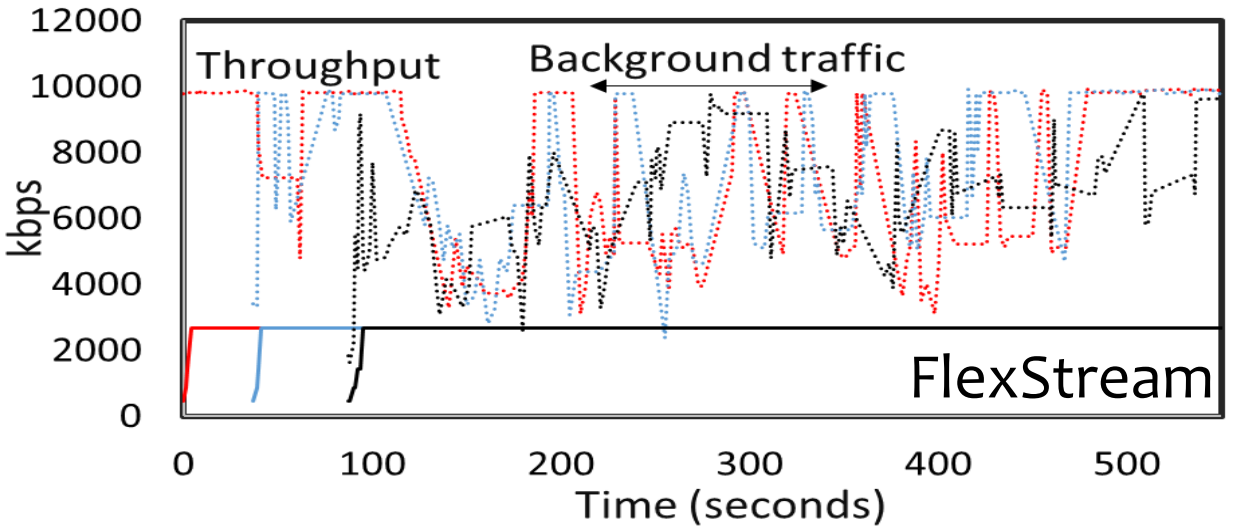
Basic Experiments

FlexStream ability to consider user priority and screen size



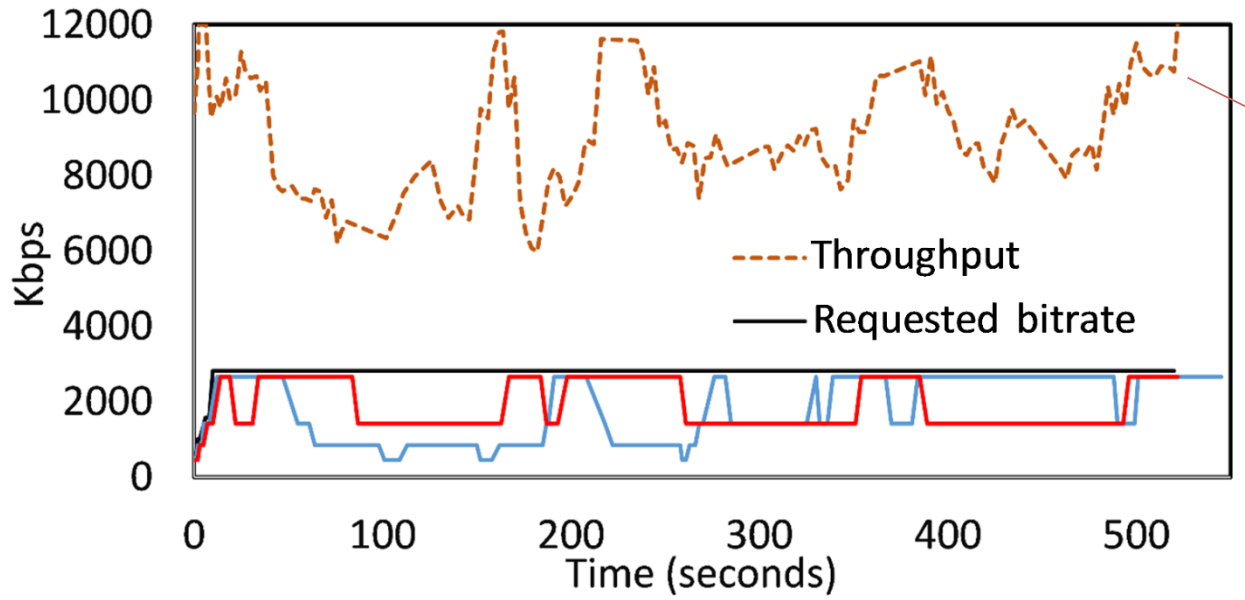
Impact of background traffic on stability with no control





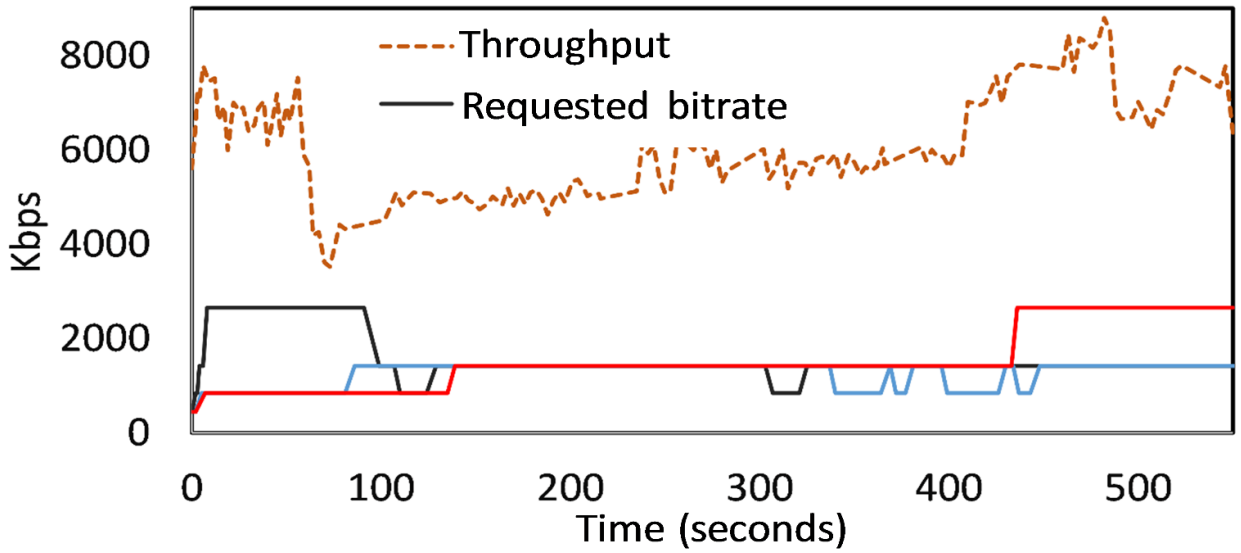
Basic Experiments – Cellular

Instability and unfairness with no control

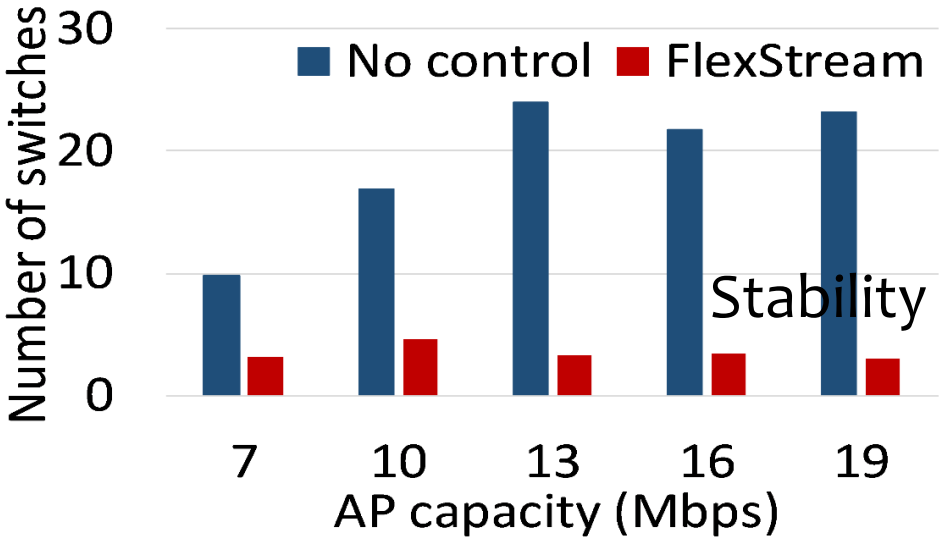


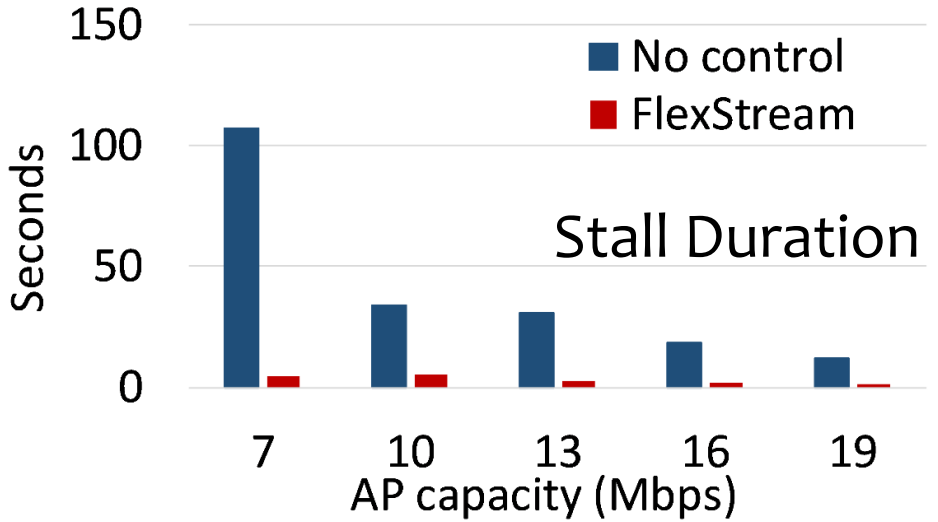
Total throughput measured by all video players

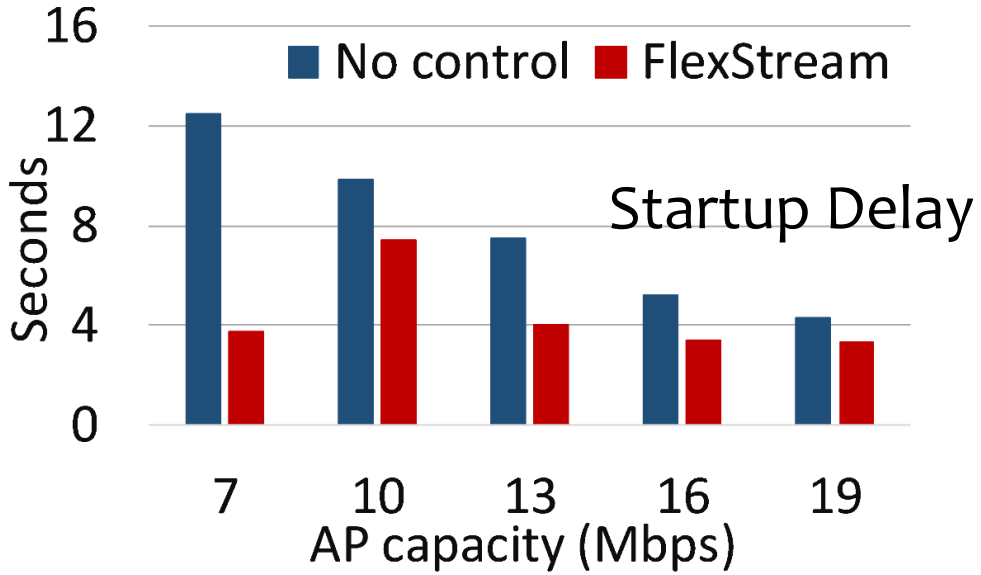
Improved stability and fairness with FlexStream

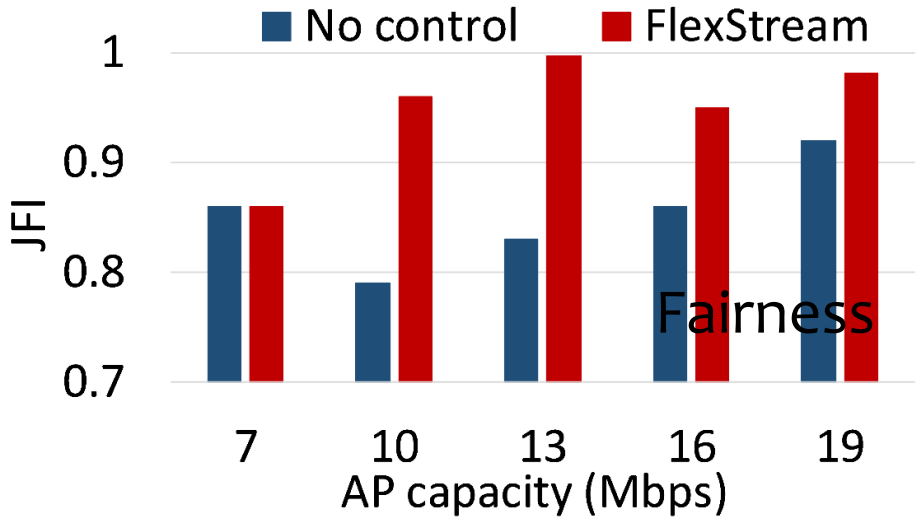


Extended Experiments









FlexStream Overheads

- GPAC player streams 1.4 Mbps video while DA is running in the background:
  - CPU utilization Overhead?
    - The CPU usage is around 1%
  - Bandwidth Overhead?
    - The total number of bytes sent and received while streaming the whole video is measured with and without enabling FlexStream.
    - FlexStream feedback and control messages found to incur less than 0.00004% of the total bandwidth needed to stream the whole video.

Conclusion and Future work

- **We introduced:**
  - **SDN-based framework** that extends SDN functionality to mobile end devices.
  - **An optimization method** to improve video QoE considering various context information, and validate it using real experiments.
  - **The first working implementation of the SDN extension** to commodity mobile devices that runs over WiFi and cellular without support from the network infrastructure.

- **Future work:** Integrating FlexStream into the existing network policy functions and obtaining link capacity and other state from the network directly.

Thank You!

QUESTIONS

tnadeem@vcu.edu

https://music.lab.vcu.edu/