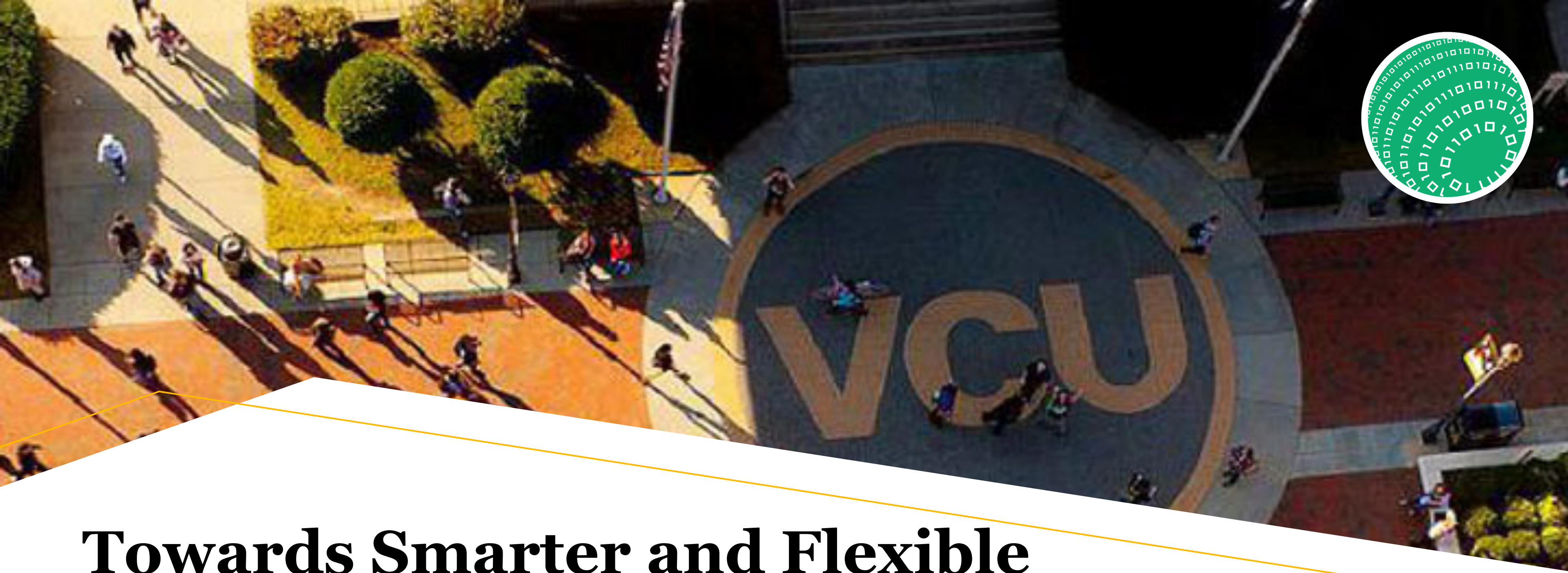


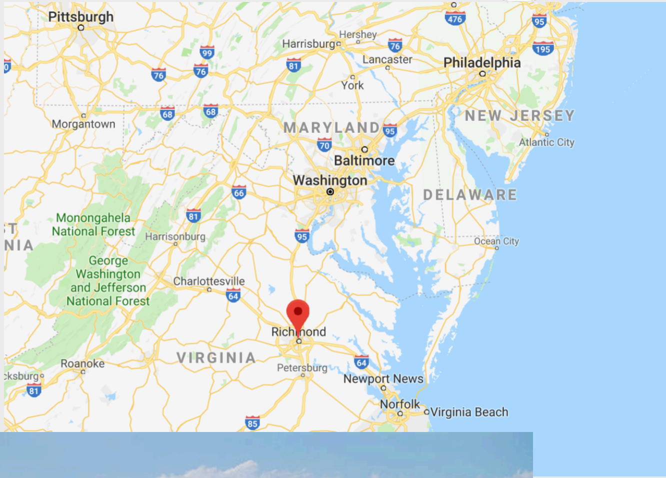
Towards Smarter and Flexible Network Edges using Extreme SDN

VCU

Tamer Nadeem
tnadeem@vcu.edu

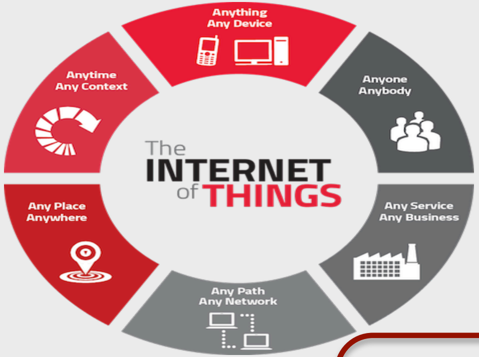
Where I Came From

- Virginia Commonwealth University (VCU) was initially established in 1838 as Medical department of Hampden-Sydney College
- School of Engineering was established in 1998
  - Became College of Engineering in 2018
- Department of Computer Science joined in 2001
- Some statistics about CS
  - 425 undergraduate students
  - 64 graduate students
  - 18 tenure/tenure track faculty (we are hiring)
  - 5 teaching faculty



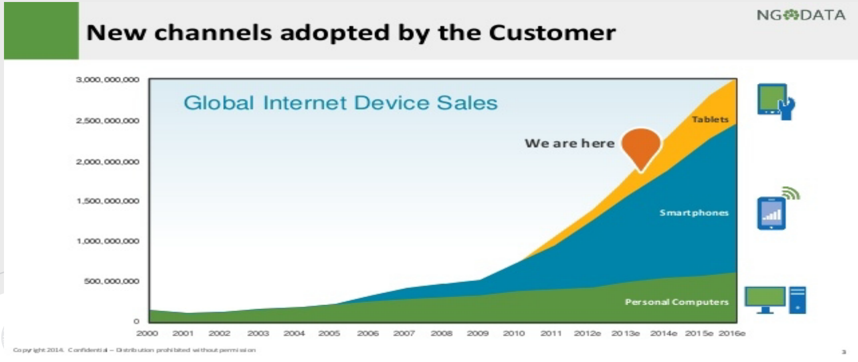
Towards Smarter and Flexible Network Edges using Extreme SDN

New Computational Era!

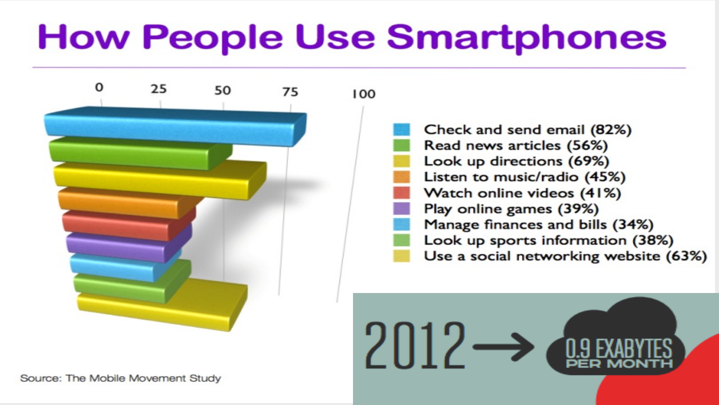


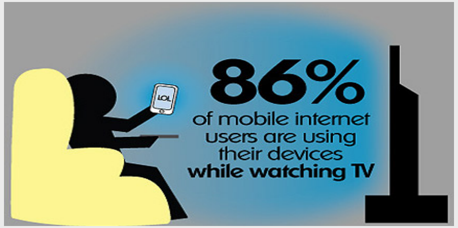
In 2017, 68% of U.S. population uses a smartphone.

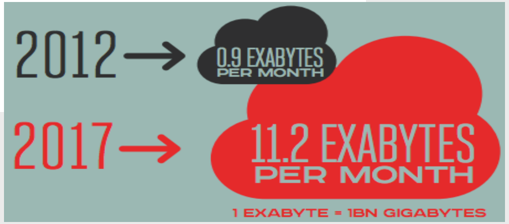
IoT connected objects are expected to reach 18 billion by 2022



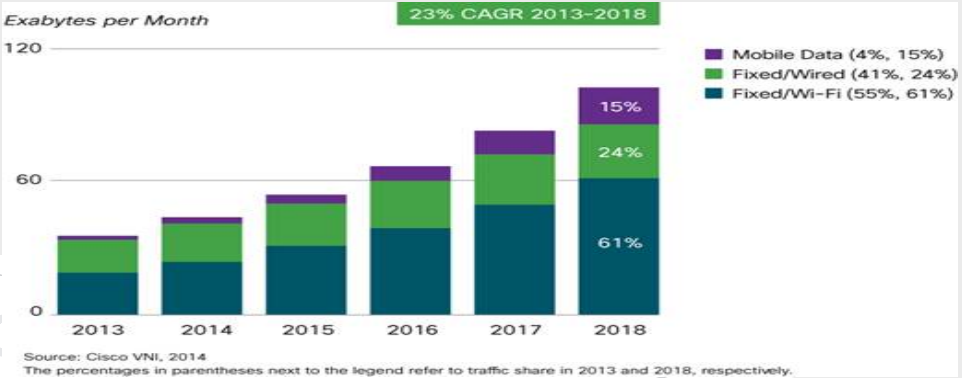
Smart Devices Usages







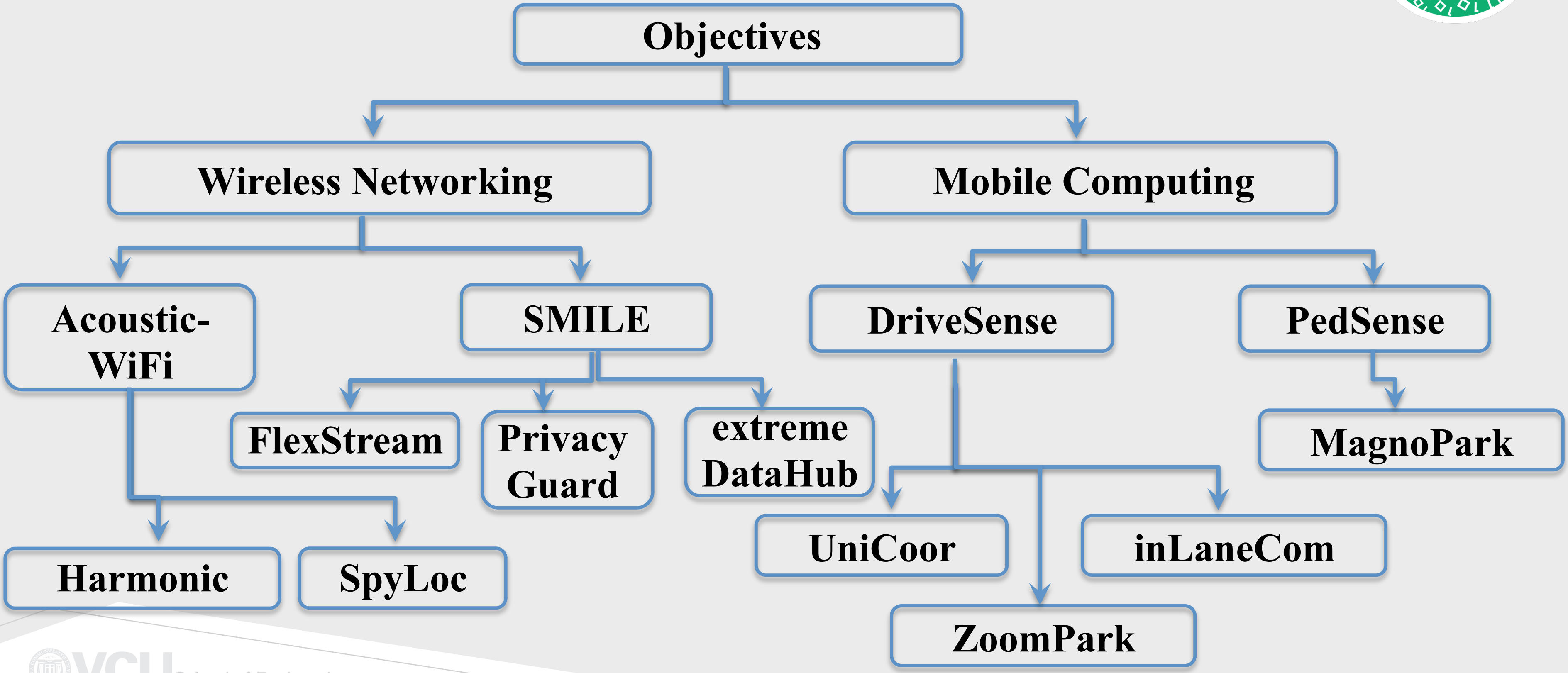
79% Of people of age 18-44 have their smartphones with them 22 hours a day



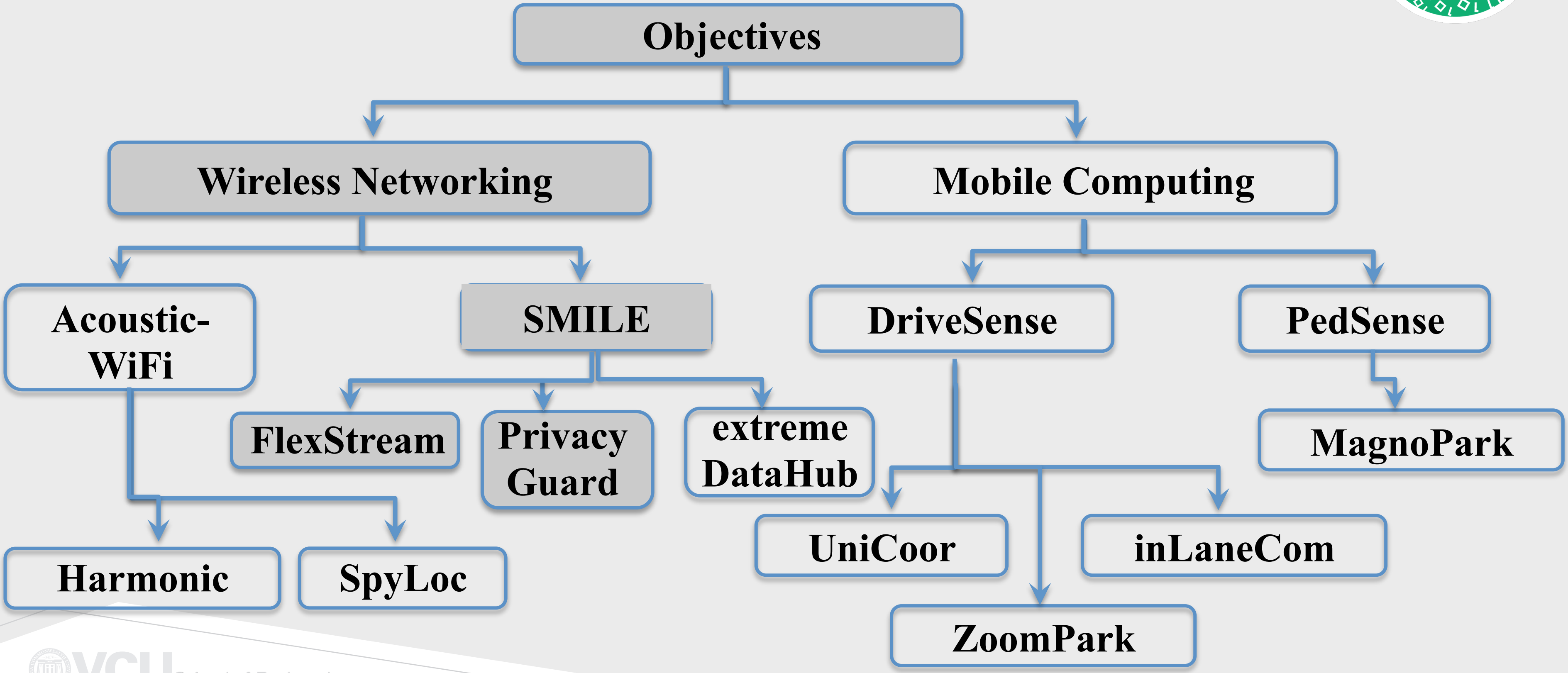
High volume of wireless traffic

74% of smartphones data goes through Wi-Fi

New Era - Research Objectives



New Era - Research Objectives



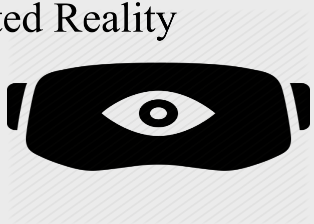
SMILE
SMart and Intelligent wireLess Edge Framework for Next Generation Networks

Rapid Growth of Mobile Data Traffic

In 2014, an average of 40,000 app were added to Apple App store a month.

Smart devices runs numerous and wide variety of applications

Virtual/Augmented Reality



Low latency/High Bandwidth new emerging applications

Best-effort Quality of Service (QoS) is no longer a satisfactory solution

Wireless Network is Complex

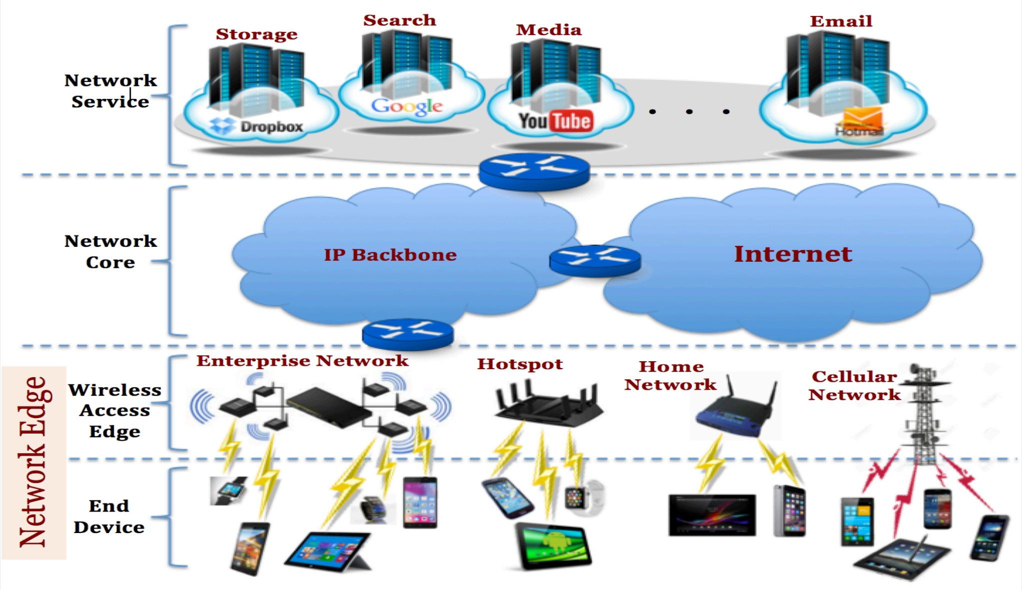
Poor utilization of wireless connectivity

Network behavior of smartphone is not smart enough

Ignores Users

Lack of control and visibility over wireless traffic from/to end-devices

Smarter Wireless Network Edge



Provide optimal performance and high quality of experience to a variety of users and applications

SDN all the way to End-devices

Wireless Network Edge

Bringing last-hop under control of SDN framework

Provide an extensible and programmable abstraction of the wireless network edges as part of the current SDN-based solutions.

SDN-like paradigm at end-devices

- Provide programmable control and monitoring capabilities over the network stack of end-device.
- Efficient interaction with the SDN-based wireless network infrastructure/Cloud.
- Provide new services and tools that can enhance the user's experience.

SDN for Wireless Network?

Recently, several approaches to adopt SDN for wireless access infrastructure

- **Cellular Infrastructure** – OpenRadio, SoftCell
- **Wi-Fi Infrastructure** - Odin, OpenSDWN, SWAN

However:

- **Targeted to solve network management problem from infrastructure point of view.**
- **Incoherent design between wireless and wire part of the network.**
- **No plan to bring the last-hop under the control of SDN framework.**
- **No intension of providing flexibility and programmability to the end-devices.**

SMILE – Objectives

Ensure QoE of end-users for both managed and un-managed wireless network environments

Less dependency on network infrastructure

Fine-grained control and reliable monitoring capability

Device context-aware network management

Coherent and simple control of both end-devices and network devices

Extending the OVS and OpenFlow protocol to support wireless Interface

Provide programmable abstraction of wireless network edge

APIs to provide flexibility and programmability of wireless network edge

Outline

- SMILE - SMart and Intelligent wireLess Edge
  - Framework
  - Data Plane – Flow Manager
  - Control Plane – Multi-Interface, TrafficVision
- SMILE – Services
  - FlexStream - Toward making wireless network edges Traffic-aware
  - PrivacyGuard - An application-aware programmable network security solution for mobile devices

Outline

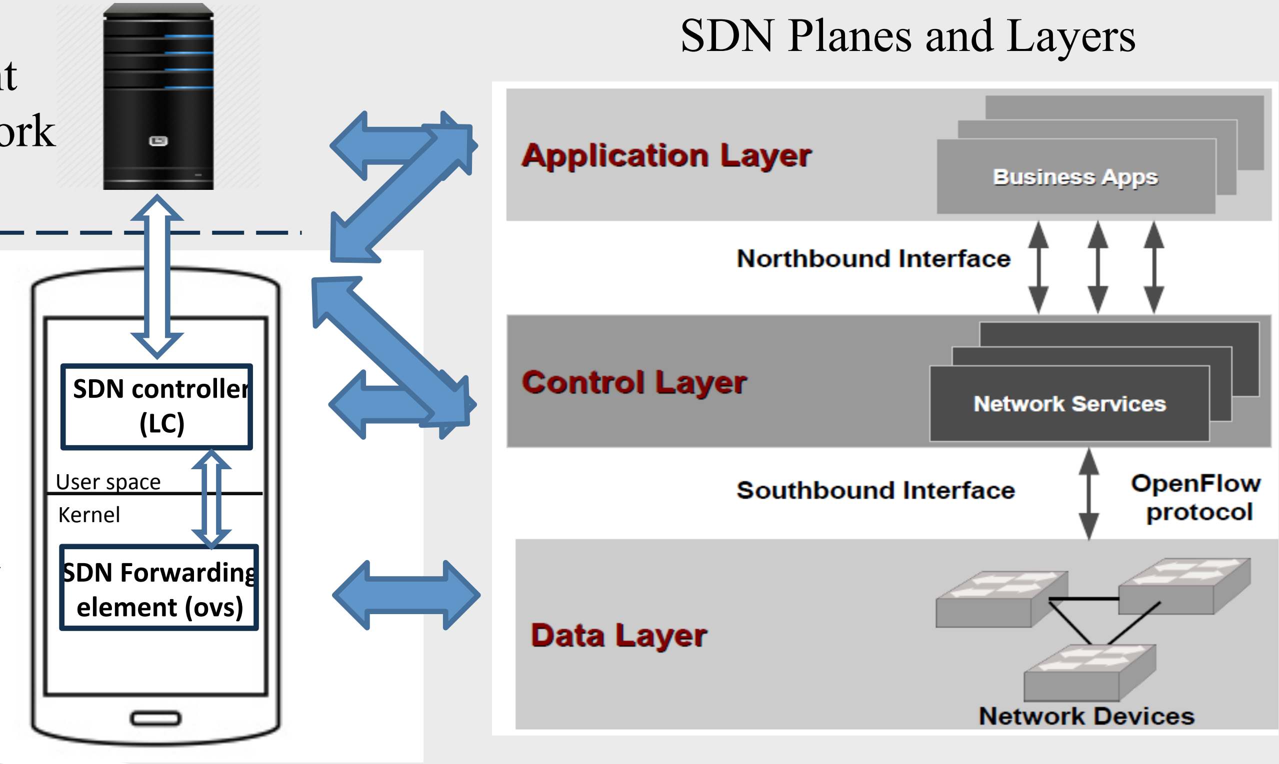
- SMILE - SMart and Intelligent wireLess Edge
  - Framework
  - Data Plane – Flow Manager
  - Control Plane – Multi-Interface, TrafficVision
- SMILE – Services
  - FlexStream - Toward making wireless network edges Traffic-aware
  - SafeEnd - An application-aware programmable network security solution for mobile devices
  - iHub - Fine-Grained Programmable Hub for BLE Internet-of-Things
- Other Projects

SMILE - SDN on End Device (*extreme SDN*)

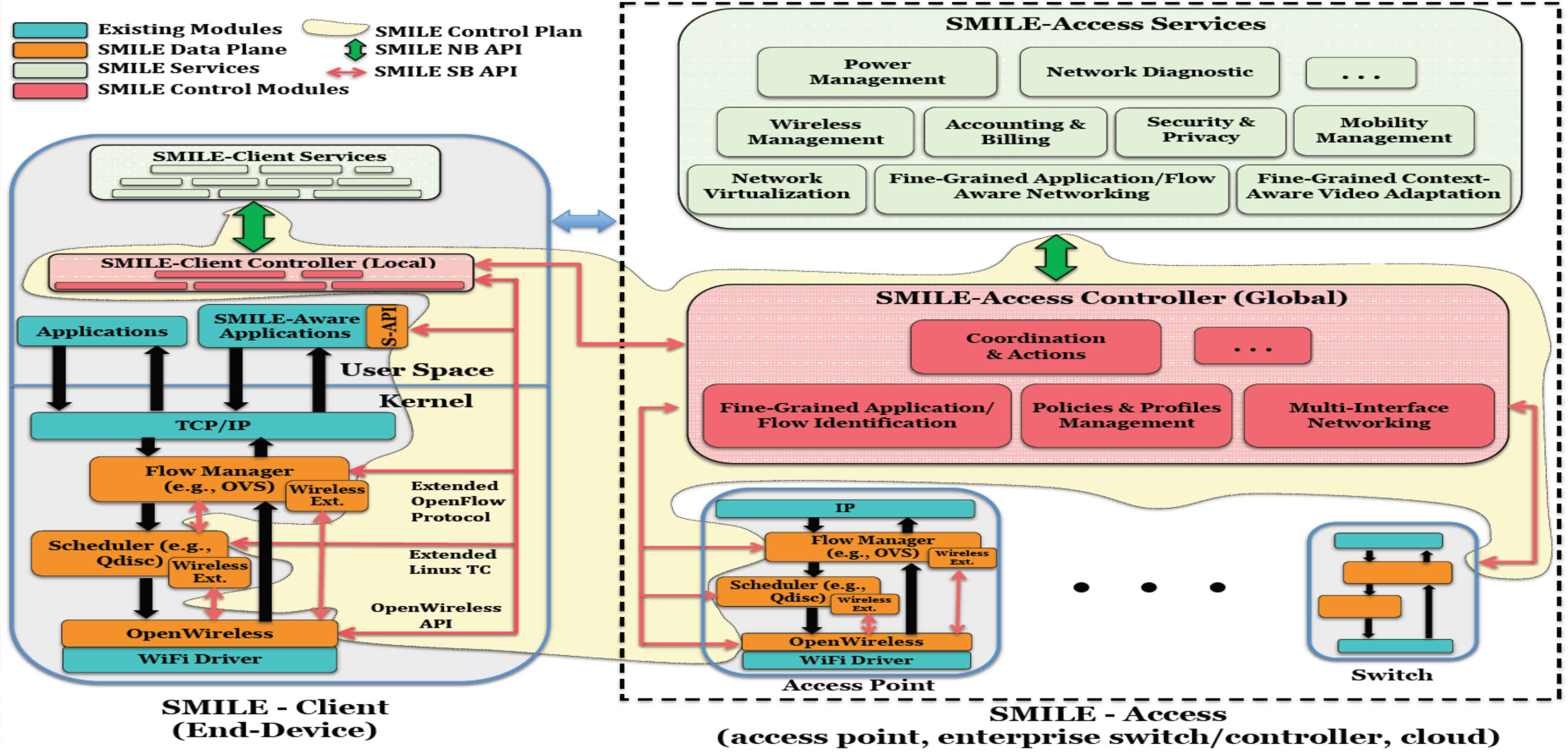
Resource management (cloud or wireless network infrastructure)

Applying network policies using SDN components on an end device.

SDN Planes and Layers

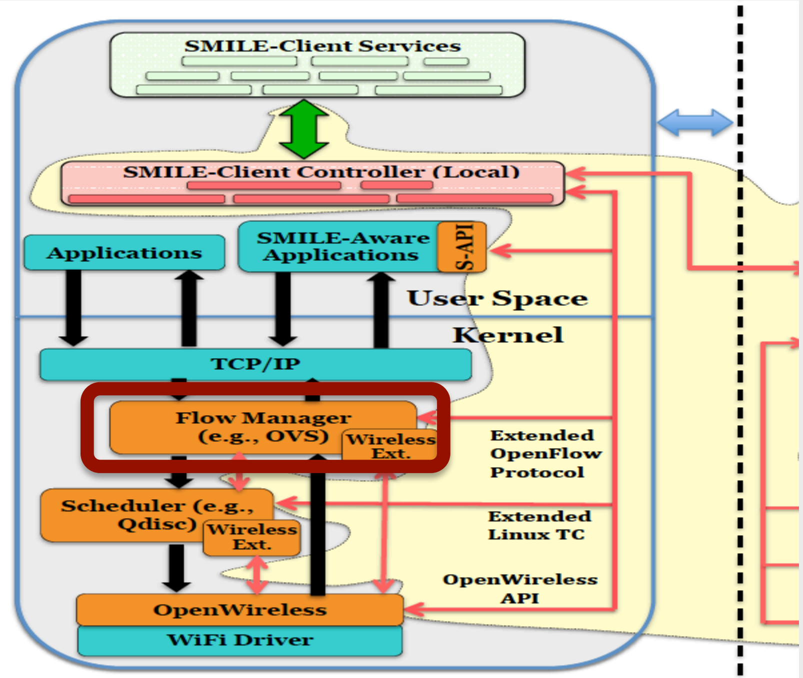


SMILE - Framework



SMILE – Data Plane - Flow Manager

FlowManger: Apply per-flow policies and and collect per-flow statistics



SMILE - Client (End-Device)

1. Leverage and extend the **OVS** to collect **per-flow statistics**, such as packet sizes, inter-arrival packet time, burst duration, throughput, and inter-burst time etc.

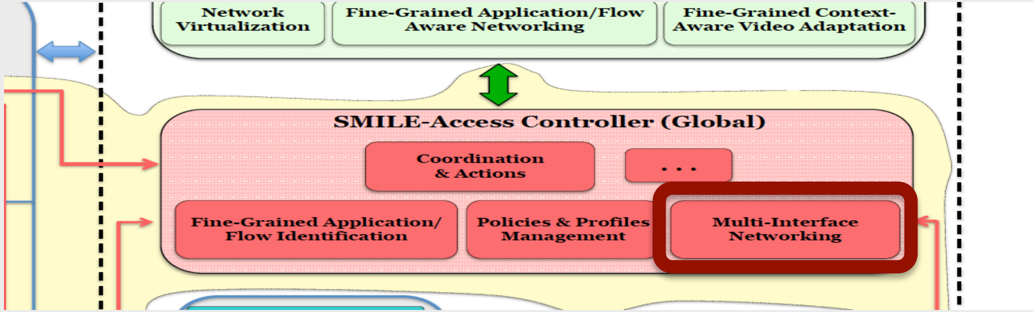
2. Leverage and extend the **OVS** to apply **per-flow policies**, such as traffic shaping, QoS marking, access control, TCP window changing etc.

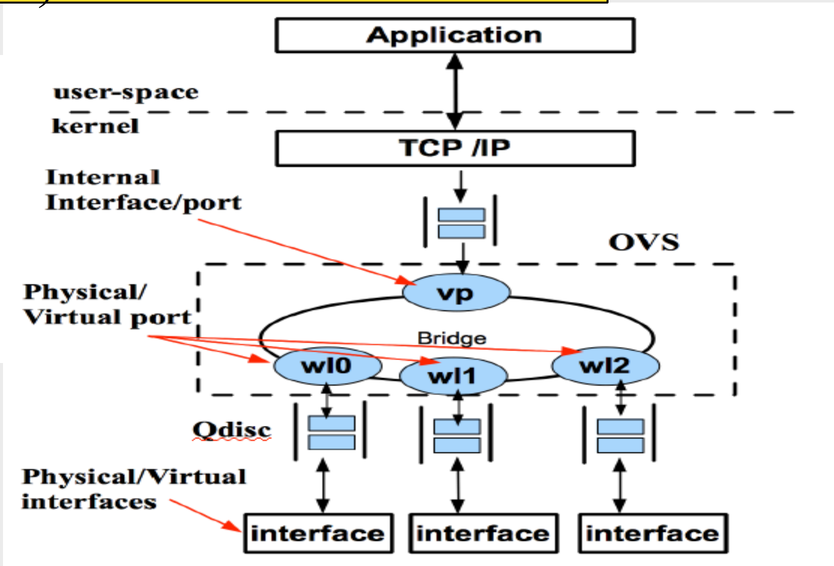
3. Leverage **XFRM framework** to apply **per-flow IPsec policy**.

4. Collect **per-client or per-flow wireless statistics** such as RSSI, data rate, TX mode and drop count.

SMILE – Control Plane – Multi-Interface

Multi-interface networking to support the multiple wireless network interfaces (Wi-Fi, LTE, 3G etc.) of smart devices

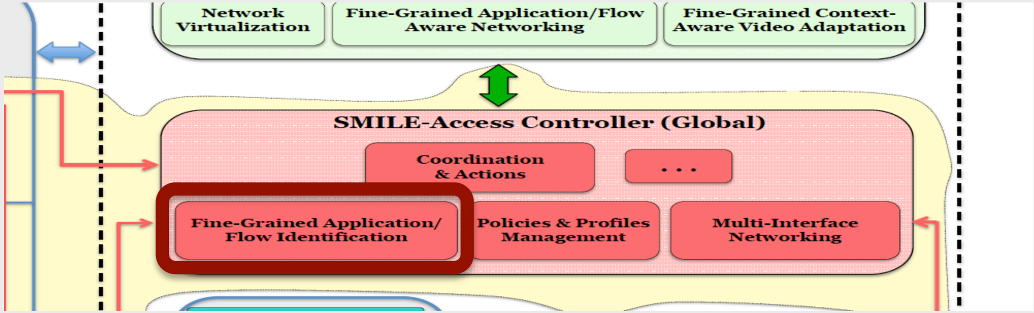




- **Layer 3 solution** to make the integration of multiple interfaces **transparent** and **seamless** to upper layers
- Leverages the OVS to create a bridge, where we add one **internal interface/port (vp)**, and a separate output port corresponding for each **physical or virtual** wireless interface

SMILE – Control Plane – TrafficVision

TrafficVision*: On-fly, light-weight and fine-grained "traffic-awareness" system

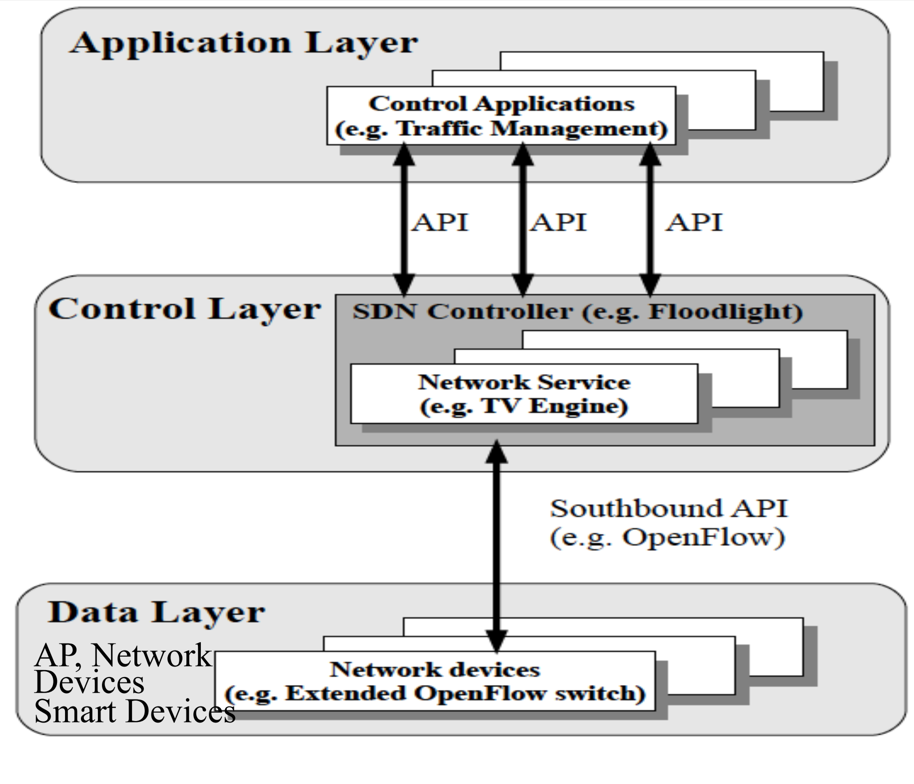


Enable **on-fly fine-grained visibility and control** over the network traffic generated by different applications and corresponding various flow-types

- Light weight and flexible **application and flow type awareness framework** for wireless network edges.
- Extract **new flow statistics** such as packet sizes, directions, sequences, and timestamps
- Provide **scalable, efficient and real-time** solutions for **classifying the network traffic flows** based on Machine-Learning (ML) techniques.

* Mostafa Uddin, Tamer Nadeem, " Traffic Vision: A case for Pushing Software Defined Networks to Wireless Edges", The 13th IEEE International Conference on Mobile Ad hoc and Sensor Systems (IEEE MASS 2016), Brasilia, Brazil, October 10-13, 2016

TrafficVision - System Overview

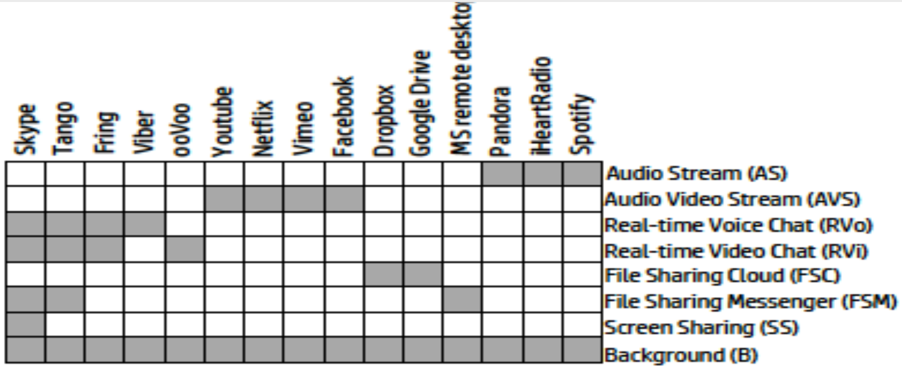


- We use standard event listener for different popular applications and flow-types.
- Send command to SDN controller according to control applications

- Aggregate flow statistics information.
- Extract features from the collected flow statistics.
- ML-classifiers to identify app and it's flow-type.
- Collect ground-truth Data

- Addition flow statistics of packet sizes, and arrival timestamps.

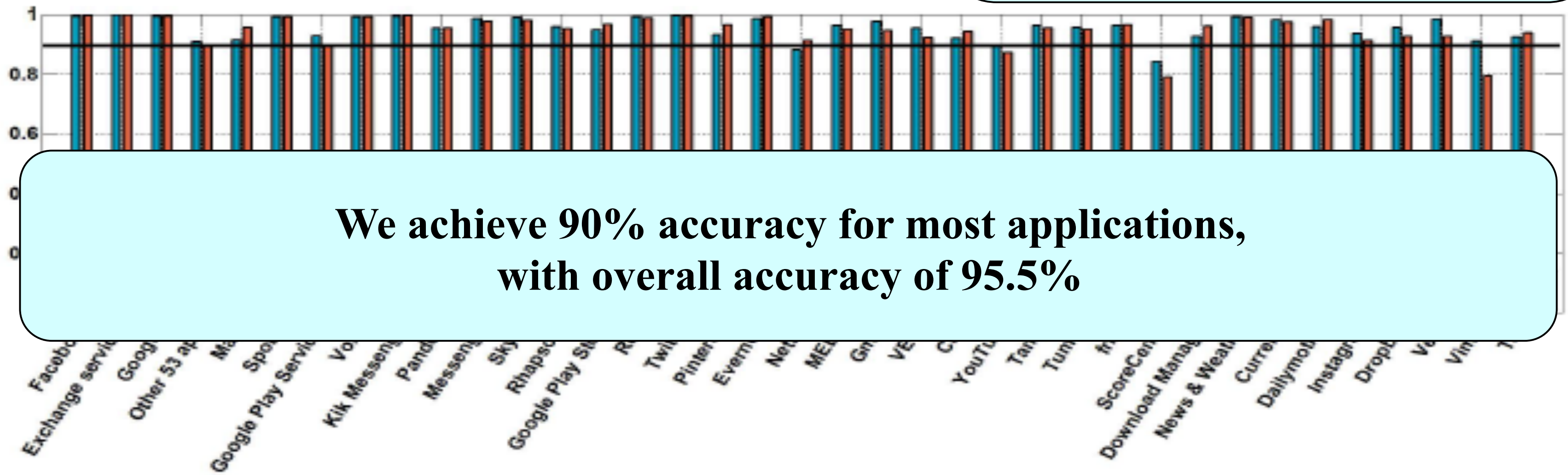
TrafficVision - Classification Accuracy



Two Datasets

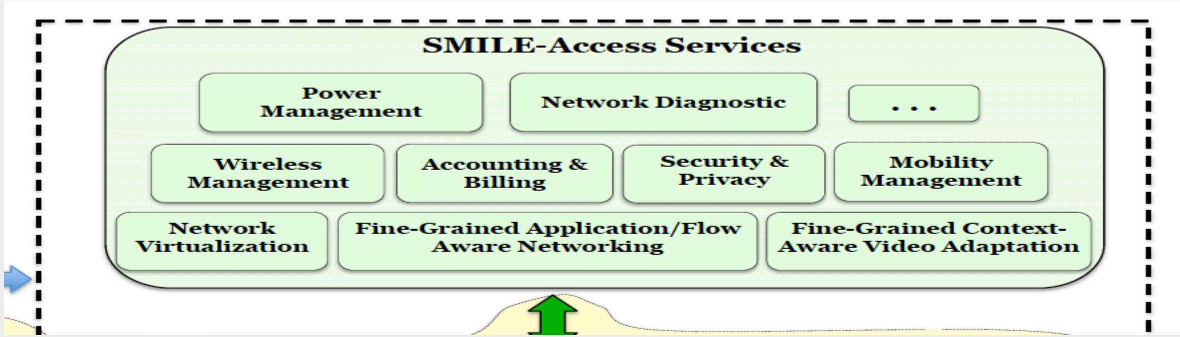
School: - Eight Volunteer
- 36 Most popular applications (Google Play)
- Over 100K flow samples

HP Labs: - 20 different applications
- 1200 flow with net duration over 10,000 min



We achieve 90% accuracy for most applications, with overall accuracy of 95.5%

SMILE – Services



WLAN virtualization enable effective sharing of wireless resources by a diverse set of users with diverse requirement

FlexStream - Edge-Based SDN Architecture for Programmable and Flexible Adaptive Video Streaming

PrivacyGuard: An Application-aware Programmable Network Security Framework for Mobile Devices

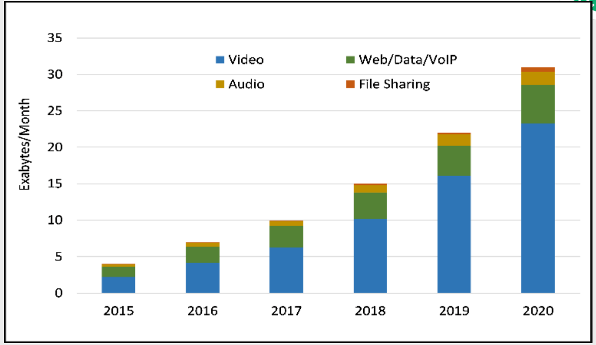
extremeDataHub: Fine-Grained Privacy-Aware Personal Hub

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

* Ibrahim Ben Mustafa, Tamer Nadeem, Emir Halepovic, "FlexStream: Towards Flexible Adaptive Video Streaming on End Devices using Extreme SDN", ACM MULTIMEDIA 2018, Seoul, Korea, 22 - 26 October, 2018

Mobile Video Traffic

- Mobile Video Traffic is dramatically increasing, by 2020 it poses 75% of the total mobile traffic*.
- HTTP Adaptive streaming protocol was adapted to improve user's QoE.
- Provide good level of QoE becomes challenging.

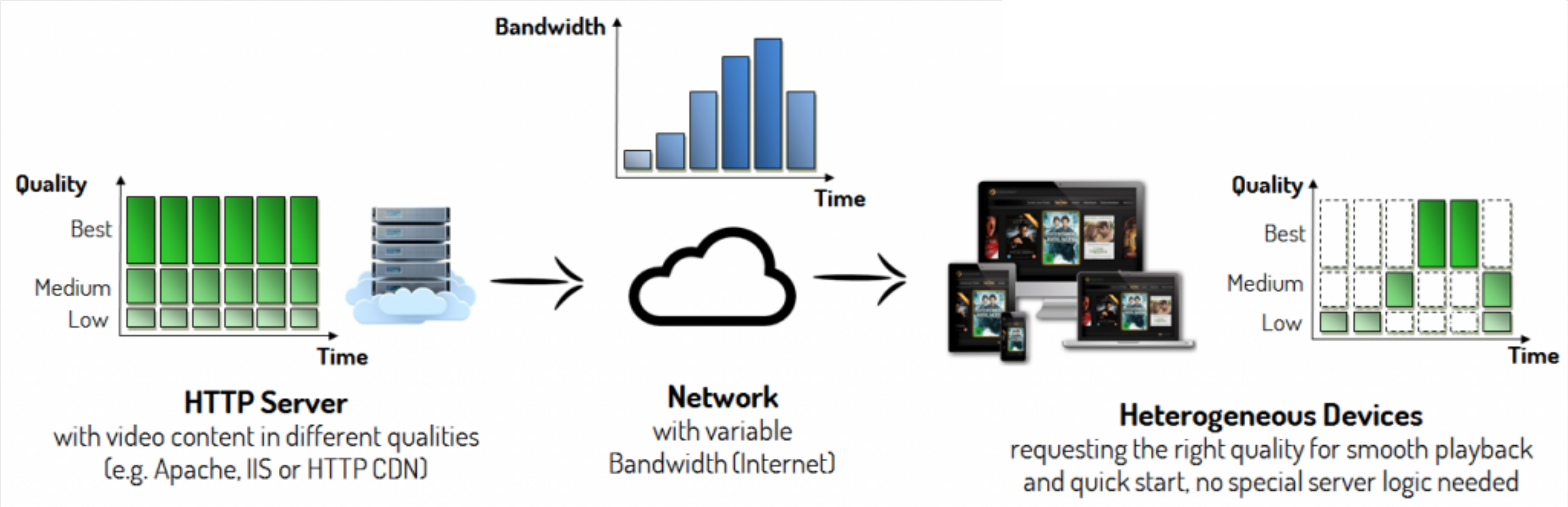


Live Streaming

Live broadcasting

* Cisco Visual Networking Index - 2017

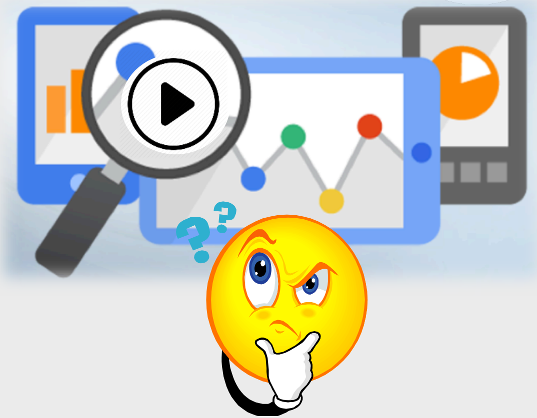
HTTP Adaptive Streaming (HAS)



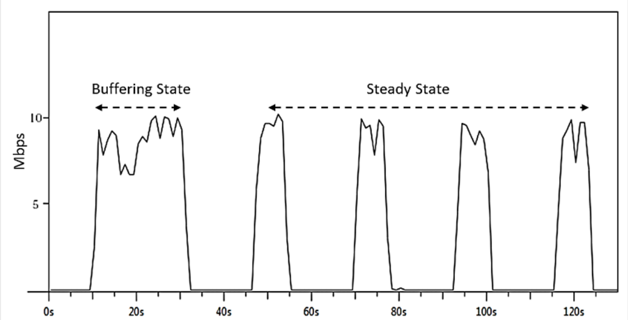
Images retrieved from: https://bitmovin.com/

Performance Issues with HAS

- When HAS players compete over the bottleneck:
  - Instability in the quality
  - Playback stalls
  - Unfairness
  - Long startup delay

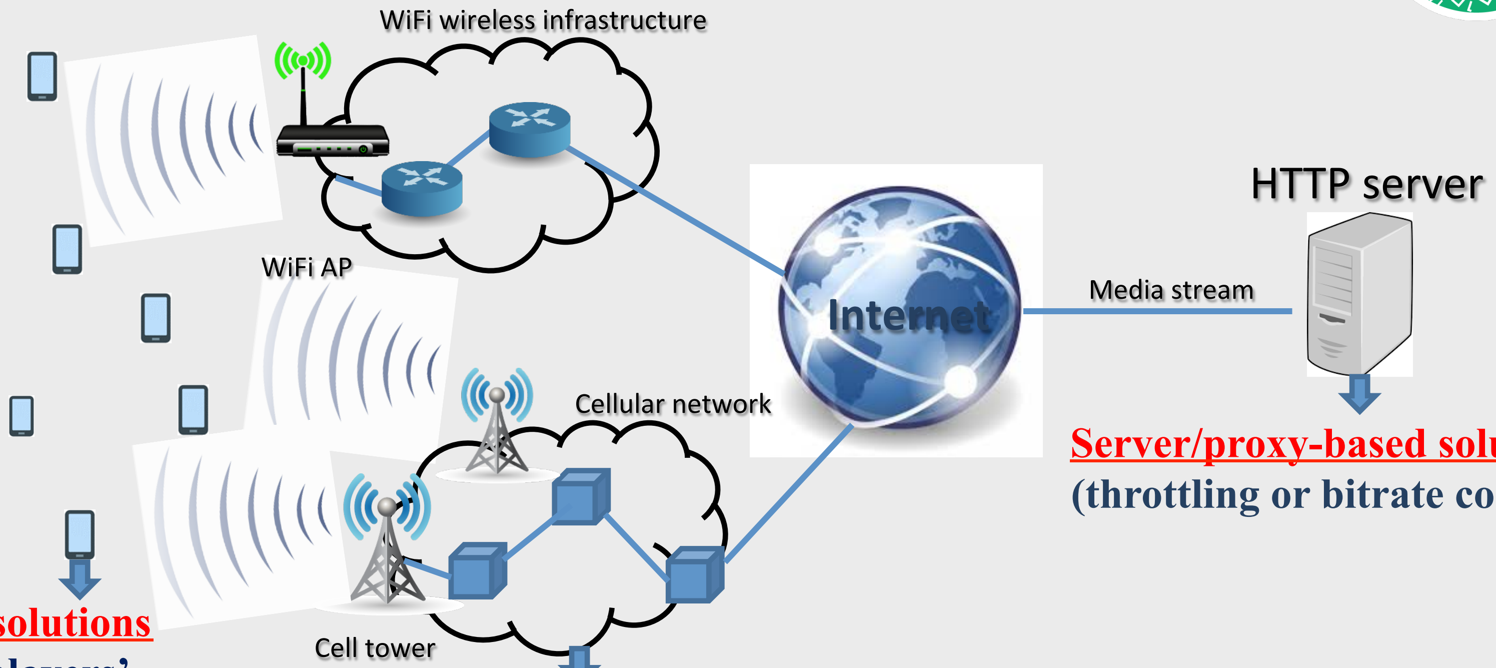


- <u>Root cause: ON/OFF traffic pattern</u>(*)



(*) 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



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

Client-based solutions
(improving players' adaptation algorithm)

Edge-based solutions
(traffic management and 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.

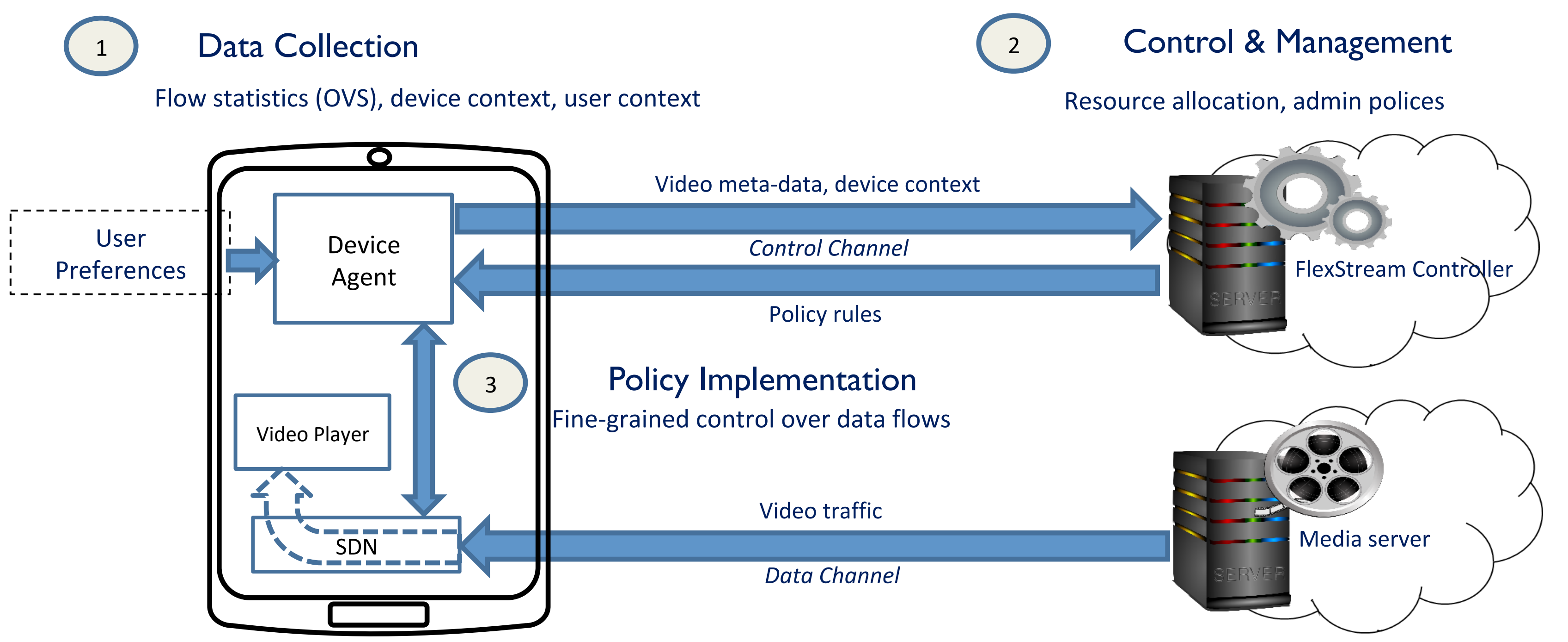
Our Solution: FlexStream

- 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

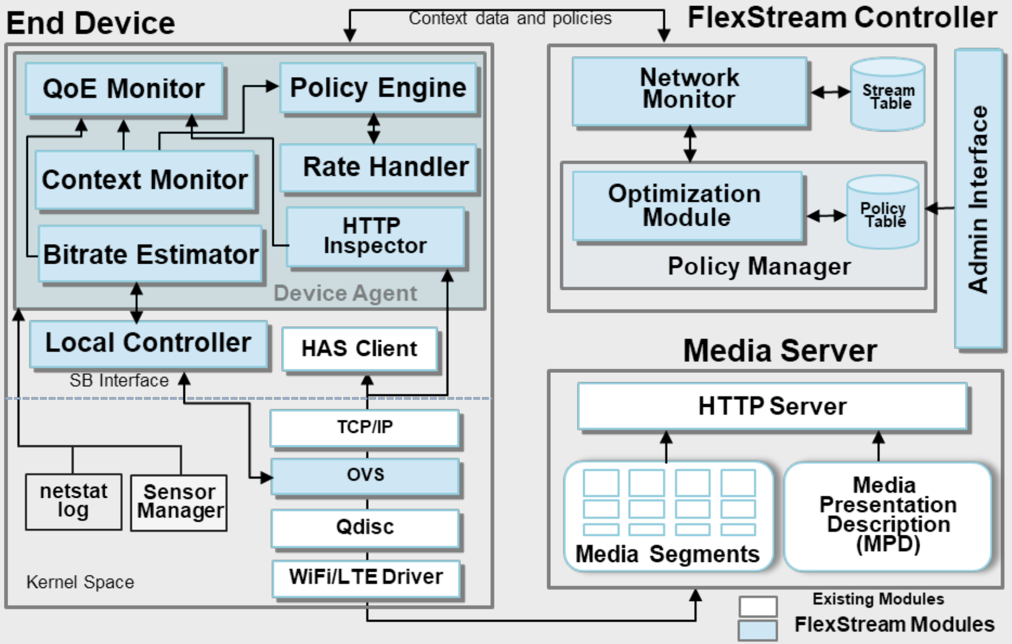
- Offloads intrusive or resource-demanding tasks from the network to end devices.
- Allows for fine-grained and intelligent management of bandwidth based on real time context awareness and specified policy.
- Flexible implementation of network policies.
- Improves video QoE:
  - Reduces quality switching 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.

FlexStream – Overview



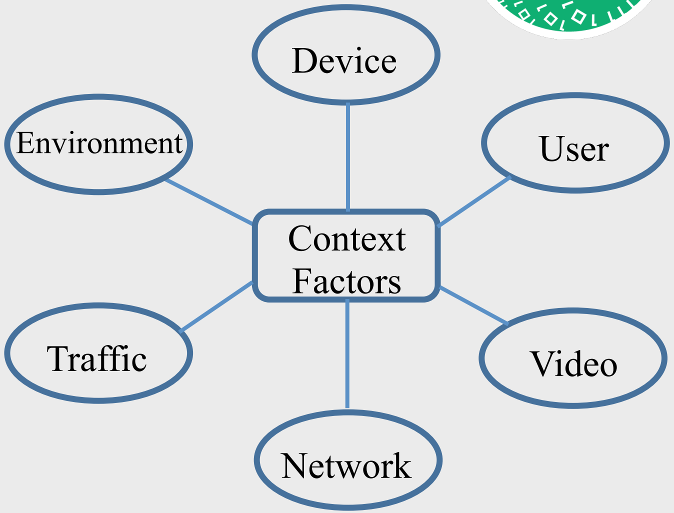
FlexStream – Architecture

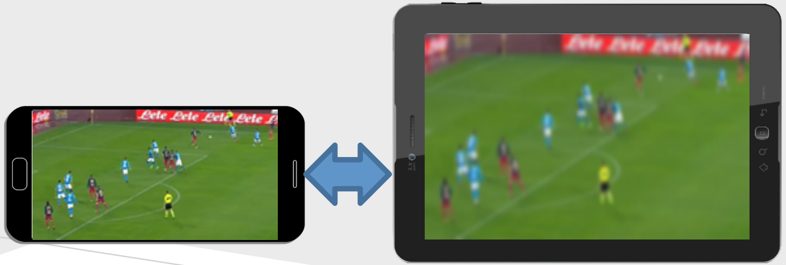
- **FlexStream Controller**
  - –Network Monitor Module: monitor the network condition through end-devices feedback.
  - –Optimization function: Allocating Bandwidth to players according to optimization policy.
- **Device Agent**
  - –QoE Monitor: reports any major drop in the throughput that would directly impact the QoE to the Global controller.
  - –Context Monitor: Monitor and report device and user context.
  - –Rate Handler: periodically measures the RTT value to the media server, calculate TCP receiving window and send it to the SDN local controller.
- **Local Controller and Data Planes (OVS)**
  - –Collecting statistics from current video streams
  - –Forcing the optimization policy received from global controller.



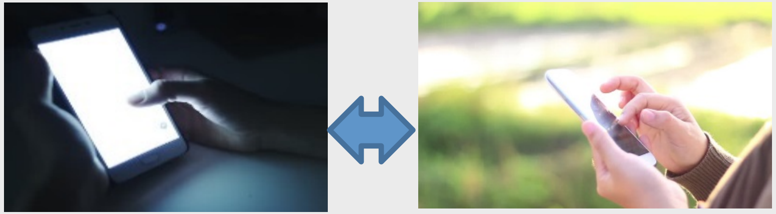
FlexStream – Context-Awareness

- 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

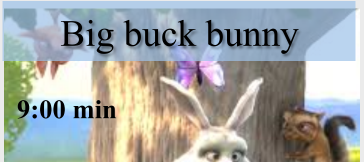
$$\delta_{ij} = \begin{cases} |r_{ij} - r_{ic}| s_i + (m - \lceil \frac{t_i}{k} \rceil), & t < t_{thresh} \\ |r_{ij} - r_{ic}| s_i, & t \geq t_{thresh} \end{cases}$$

Evaluation

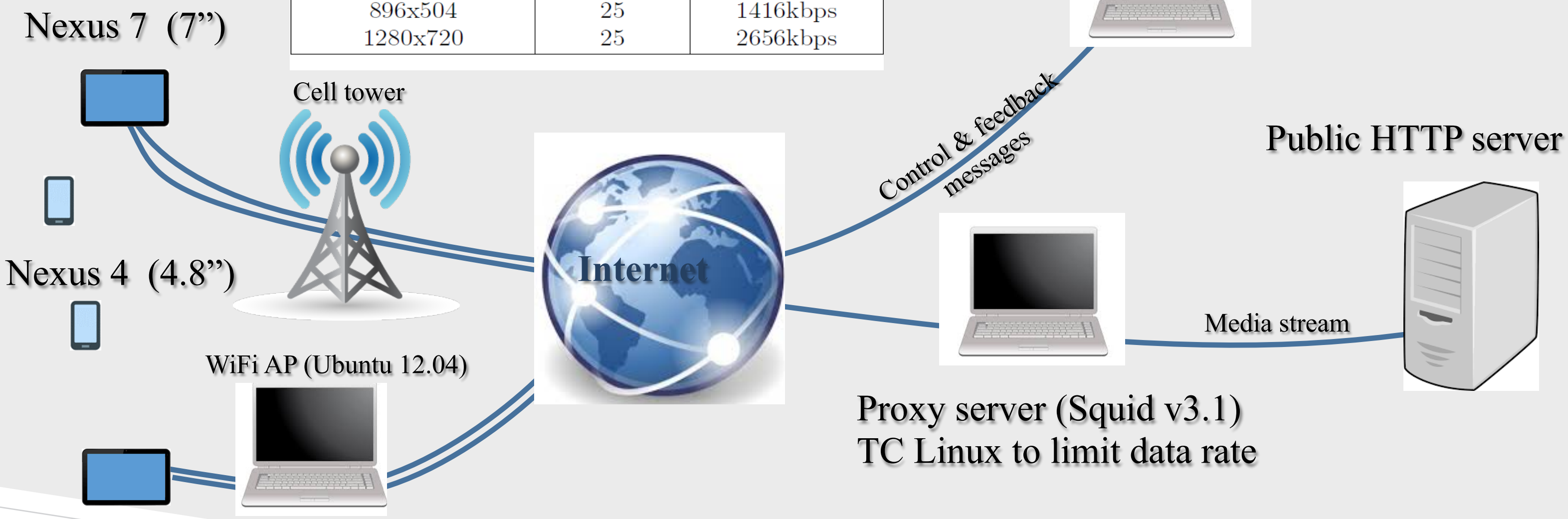
- **Quality Metrics**: Stability, fairness, stalls, and startup latency.
- **Scenarios:** Static Bandwidth and Dynamic Bandwidth
- **Experiments**
  - Basic: 3 real players in a real network.
  - Extended: 12 emulated players & server, real network.
- **Context:** User priority, screen size, link condition,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



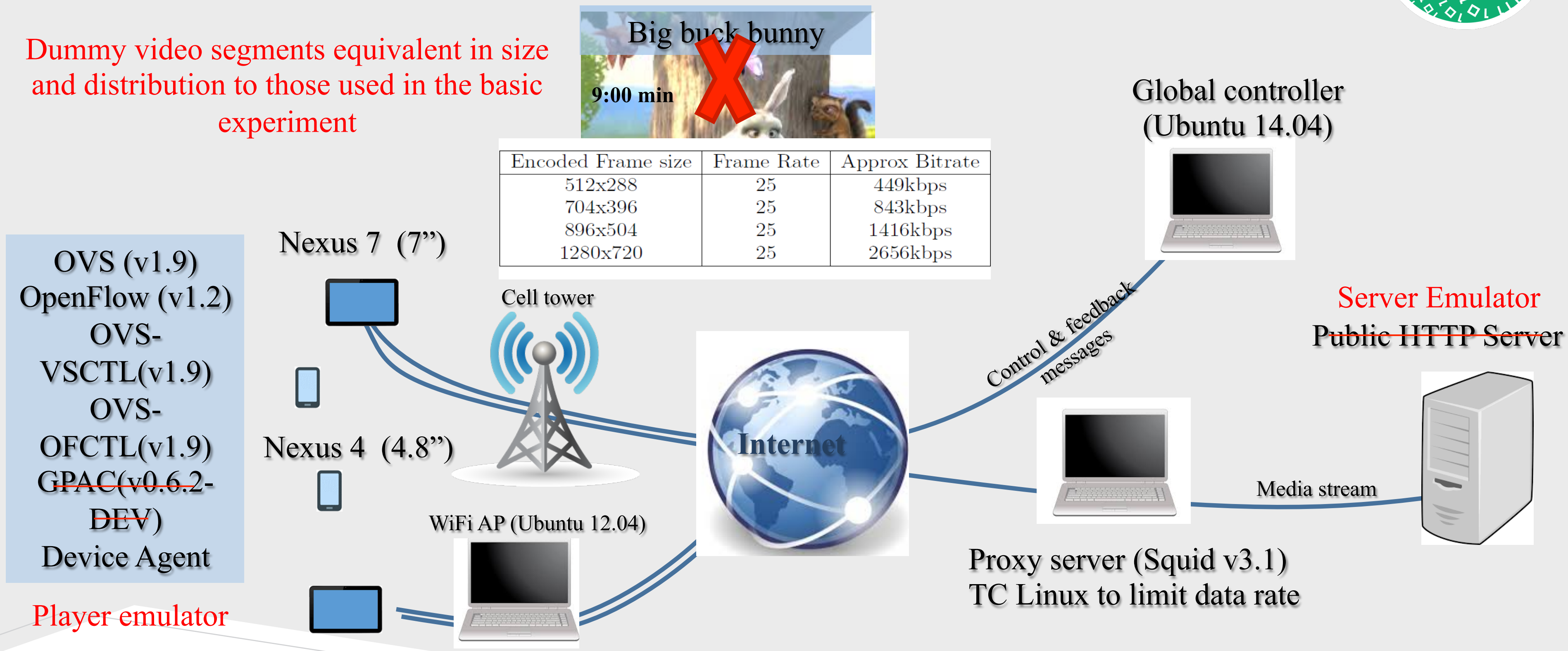
Encoded Frame size	Frame Rate	Approx Bitrate
512x288	25	449kbps
704x396	25	843kbps
896x504	25	1416kbps
1280x720	25	2656kbps



Setup for Extended Experiments

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

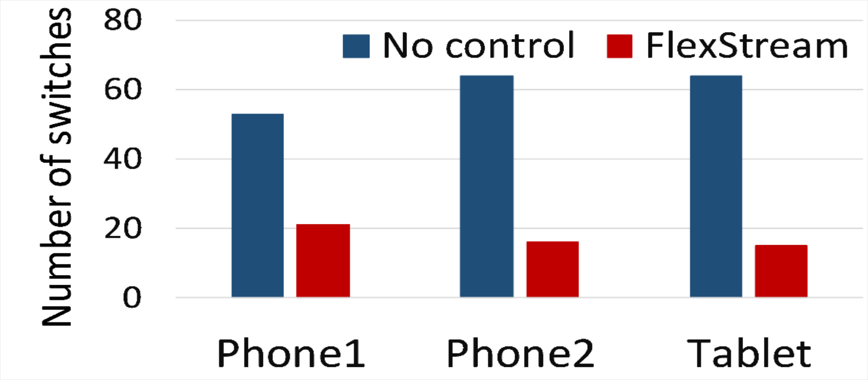
Encoded Frame size	Frame Rate	Approx Bitrate
512x288	25	449kbps
704x396	25	843kbps
896x504	25	1416kbps
1280x720	25	2656kbps



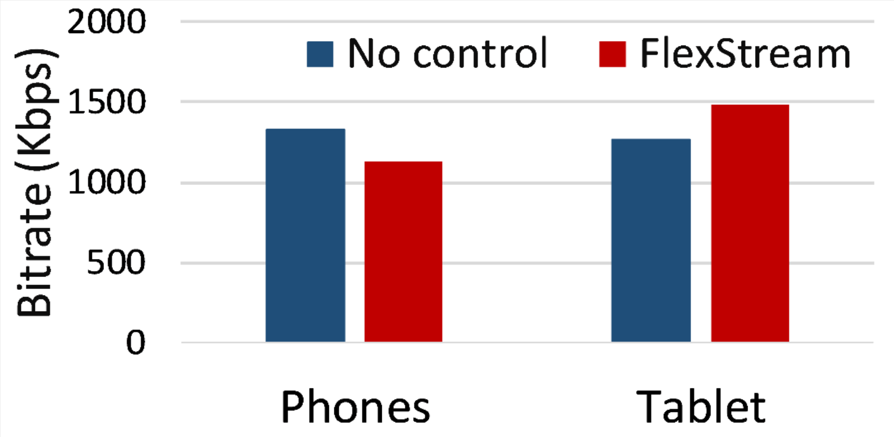
Basic Experiments

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

Average bitrate switches per device

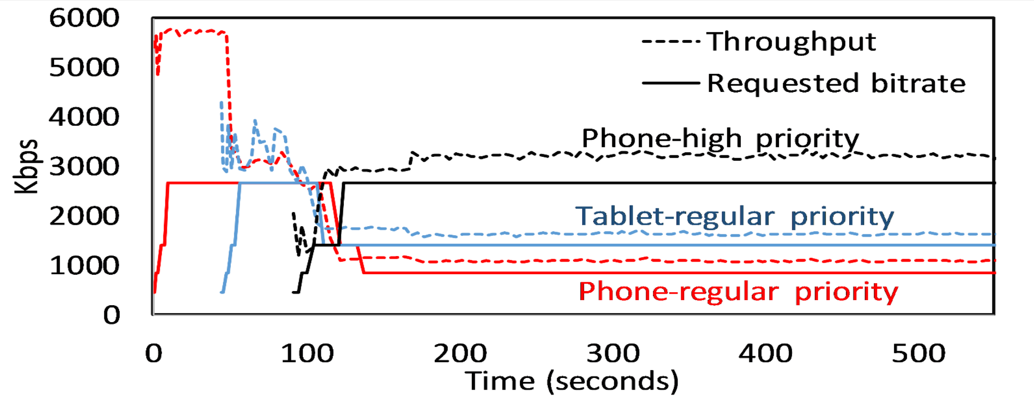


Balanced QoE for phones and tablet

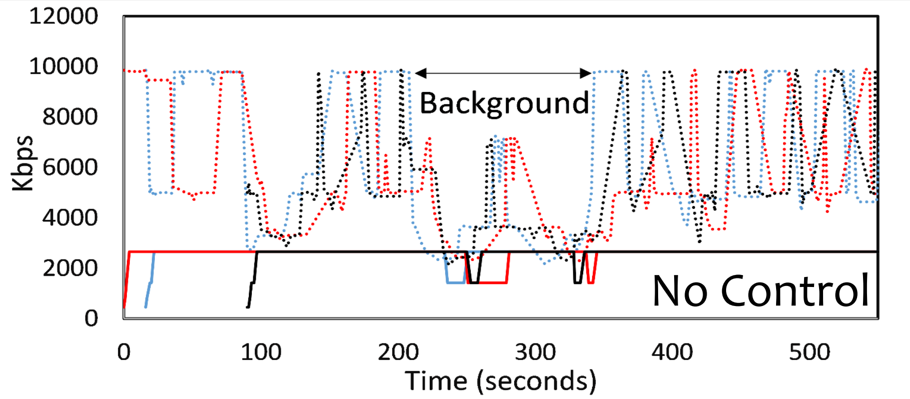


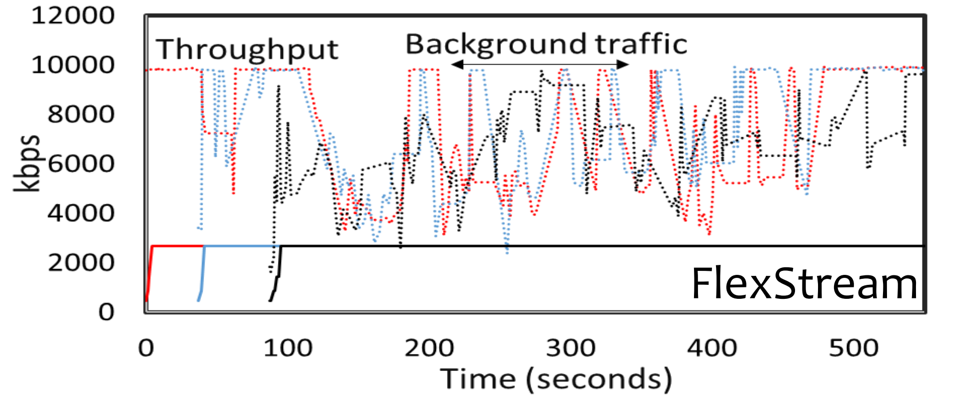
Basic Experiments

FlexStream ability to consider user priority and screen size



Impact of background traffic on stability with no control

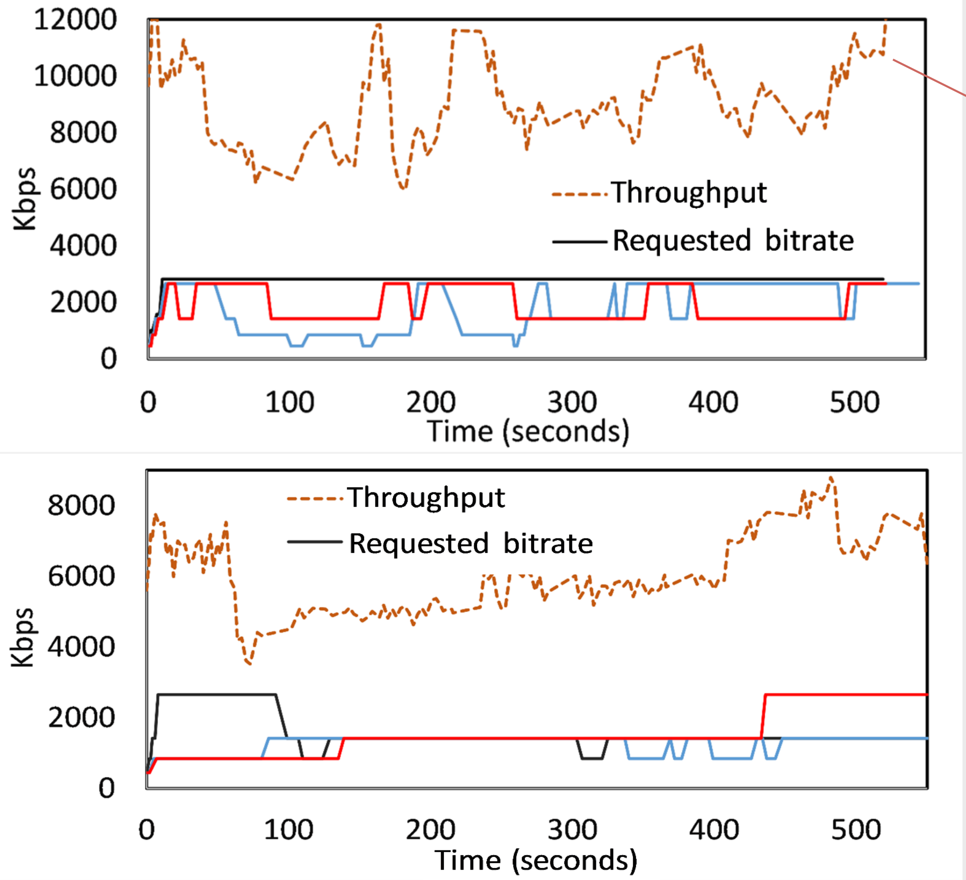




Basic Experiments – Cellular

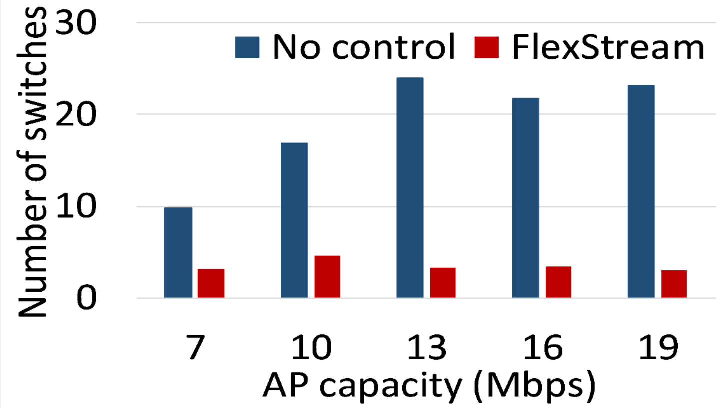
Instability and unfairness with no control

Improved stability and fairness with FlexStream

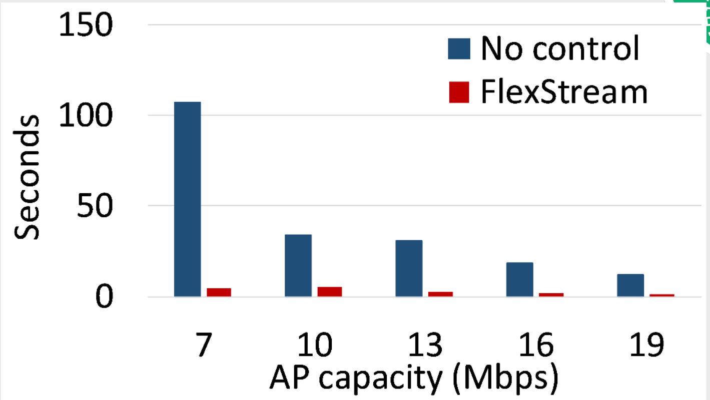


Total throughput measured by all video players

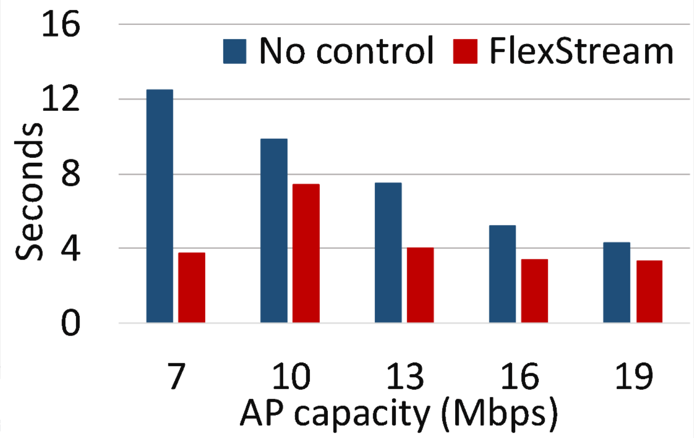
Extended Experiments



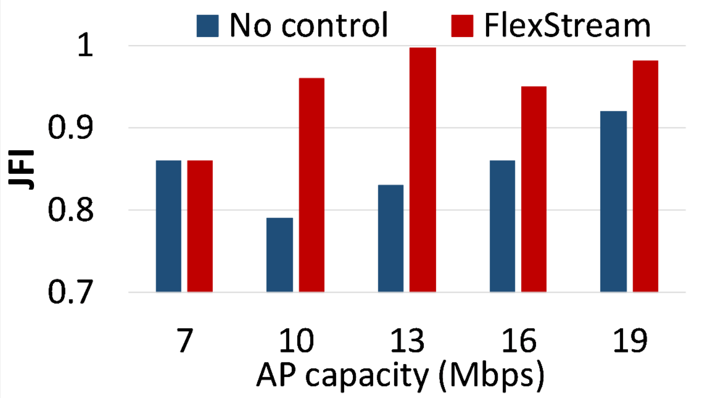
Stability



Stall Duration



Startup Delay



Fairness

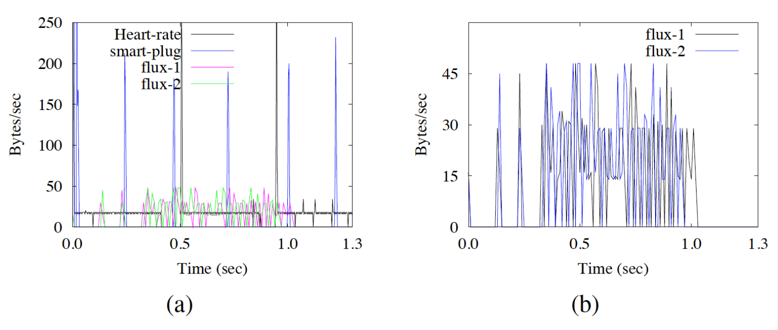
FlexStream Overheads

- 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.

PrivacyGuard: Towards Flexible Edge Privacy Framework for IoT and Mobile Applications

Introduction

- Mobile devices mostly use WiFi networks as the prominent network interface to the Internet
  - Wi-Fi networks are expected to carry almost 60% of smartphone and tablet data traffic by 2019
- Even with Wi-Fi encryption, statistical analysis of side-channel information of WLANs traffic could infer several user-related information
  - The traffic analysis of major commercial IoT devices is found vulnerable to activity inference such as user presence, device interaction and appliance usage
- Figure shows example of traffic of four IoT devices in which different IoT devices could be uniquely distinguished



Existing Solution for Sensitive Apps

Mainly focus on device/app data control and protection

- Samsung KNOX, Android for Work
- Mobile Device Management

Existing solution require infrastructure supports

- Don't support dynamic of mobile devices

Coarse-grained security policies

- Application-aware or context-aware security policy is not possible

User's are not in control of their traffic

- No flexible and user-friendly tools to meet their requirement
- Not transparent to the application

Limited work on addressing the eavesdropping attack

Traffic Obfuscation

- The most popular techniques are based on traffic shaping like traffic padding, faking superfluous packet and chopping packets into fixed size segments, and traffic morphing.
- The performance of these traffic shaping techniques in terms of efficiency and overhead varies depending on their configuration parameters.
  - For example, the efficiency of the traffic padding approach in obfuscating the traffic signature increases with the percentage of traffic packets to be padded.
  - However, this higher efficiency comes with higher overhead in terms of network bandwidth and power consumption since more bits are transmitted.
- Therefore, the configuration of these approaches should be flexible in adapting to the different context of the user needs, device characteristics, application requirements, and network conditions.
  - In addition, the approach need to be transparent (i.e., application independent).

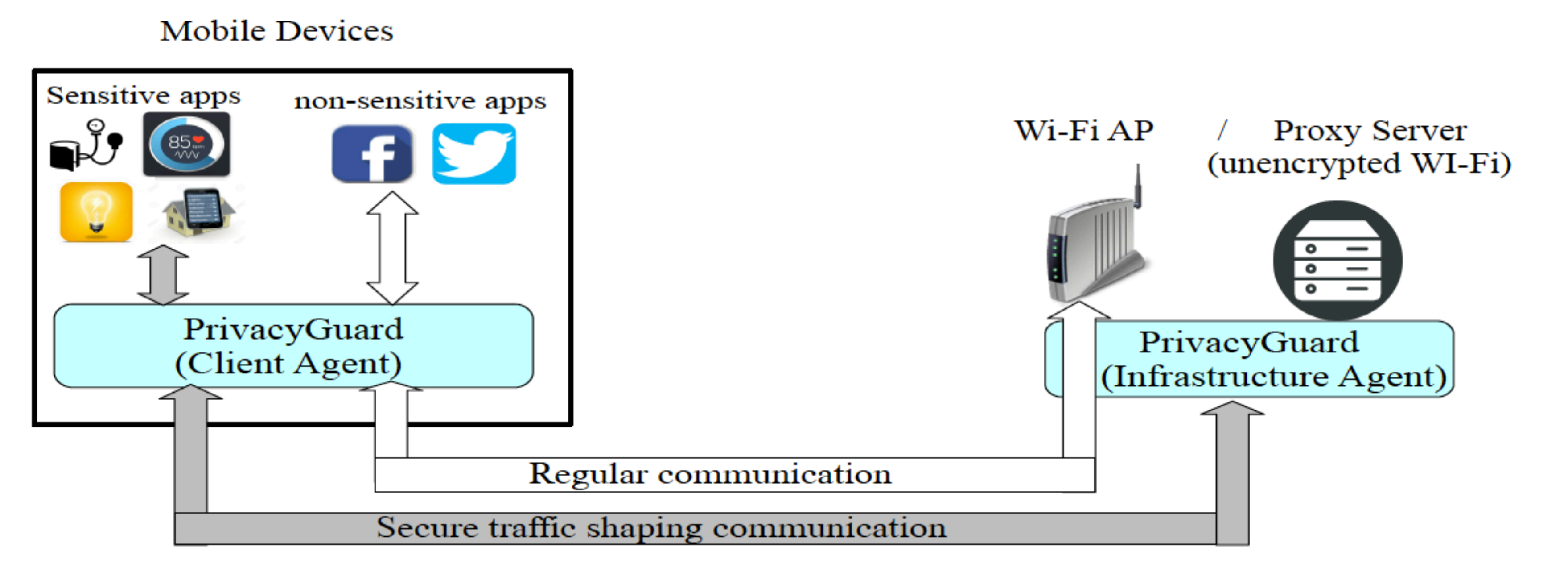
PrivacyGuard

- Flexible in applying different privacy preserving schemes to different applications and flows within applications.
  - Ex: Dropbox generates two flows for uploading/downloading a file, where in one direction data packets uses maximum possible size, while the other direction contains just identical TCP ACK frames.
- Support programmable APIs to dynamically define and configure different schemes
- Adapt to applications, users, devices, and network conditions and characteristics (context) in selecting in real-time the optimum scheme for individual applications/ flows
- Seamlessly support any application without requiring any modification on either client or server-side of the application.

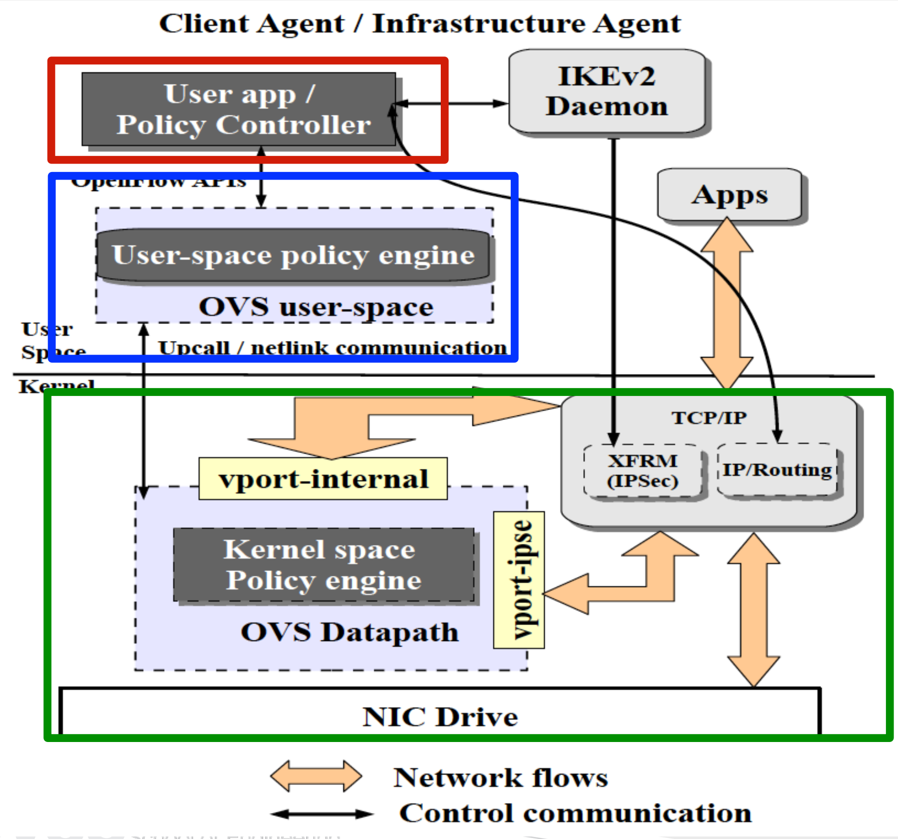
PrivacyGuard - Schemes and Contexts

- Traffic shaping schemes
  - Packet padding
    - Packet padding probability (p).
    - Padding size distribution (e.g., Gaussian, Poisson)
  - Packet delay
    - Inter-packet transmission distribution (i.e., Gaussian)
  - VPN for unencrypted traffics
- Context Information
  - Application: sensitivity level, real-time, …
  - User: location, time, …
  - Device: battery level, computing power, …
  - Network: public, load, …

PrivacyGuard – Operation



PrivacyGuard – Architecture



PrivacyGuard Controller: Convert application-aware policies to the flow-level policies.

PrivacyGuard OVS user-space: Set traffic shaping policies for the new network flows.

PrivacyGuard kernel-space

Applying traffic shaping policy before the IPsec policy

- Randomize traffic shaping policies uses IP option header.
- Many routers block packet with unknown IP option header.
- IPsec tunneling will hide the IP option header

PrivacyGuard – Flow Policy Table

Policy #1
  ID: srcIP='A', srcPort='i', dstIP='B', dstPort='j'
  CONTEXT: Location='Home' AND Time=[10PM-12AM]
  ACTION: Padding='Normal:μ=400,σ=100, p=1.0'

Policy #2
  ID: srcIP='A', srcPort='k', dstIP='B', dstPort='l'
  CONTEXT: Battery=High II Location=HotSpot
  ACTION: Padding='Normal:μ=400,σ=100, p=1.0',
      Delay='Uniform:min=0,max=20ms', IPSec

Policy #3
  ID: srcIP='C', srcPort='m', dstIP='D', dstPort='n'
  CONTEXT: Battery=Low OR WiFi Load=High
  ACTION: Padding='Normal:μ=400,σ=100, p=0.6'

Policy #4
  ID: srcIP='C', srcPort='m', dstIP='D', dstPort='n'
  CONTEXT: Battery=High OR WiFi Load=Low
  ACTION: Padding='Normal:μ=400,σ=100, p=1.0'

Evaluation

- **Configuration:** Nexus 4 with Android 4.4 (client), Linux based laptop (AP), Eight commercially available IoT devices using the client as a gateway.
- **Traffic shaping schemes:** Norm_Pad, Norm_Pad_Delay, Max_Pad_Delay
- **Metrics:** accuracy, precision, network overhead, energy overhead

Traffic Shaping Schemes Efficiency

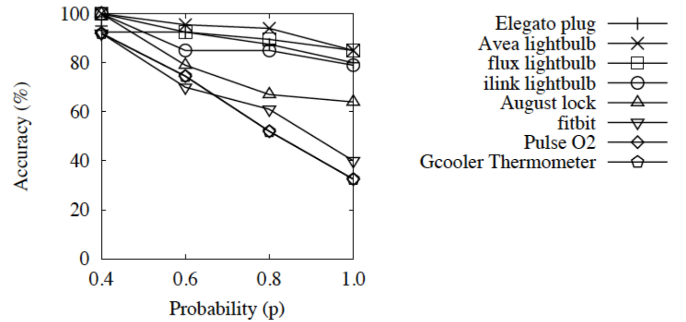


Fig. 6: The accuracy of Norm_Pad scheme for different applications and p.

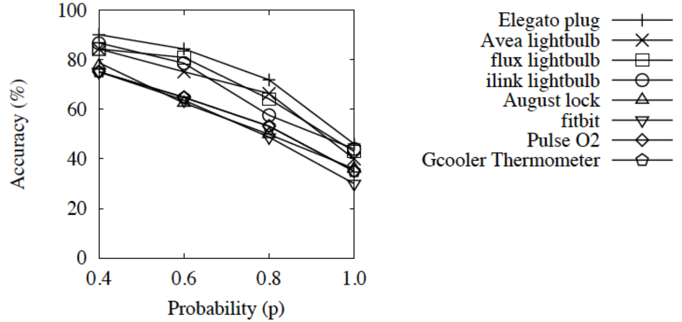


Fig. 7: The accuracy of Norm_Pad_Delay scheme for different applications and p.

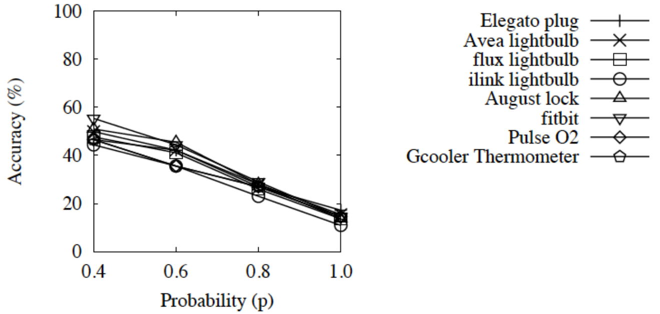


Fig. 8: The accuracy of Max_Pad_Delay scheme for different applications and p.

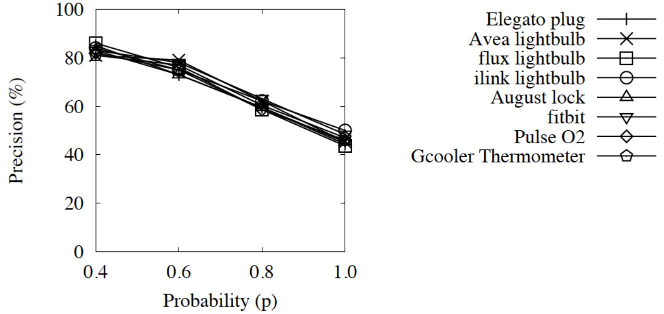


Fig. 9: The precision of Max_Pad_Delay scheme for different applications and p.

Traffic Shaping Schemes Overhead

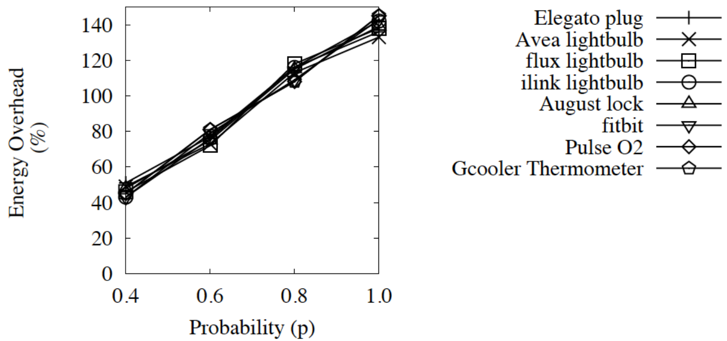


Fig. 10: The power consumption overhead of Max_Pad_Delay scheme.

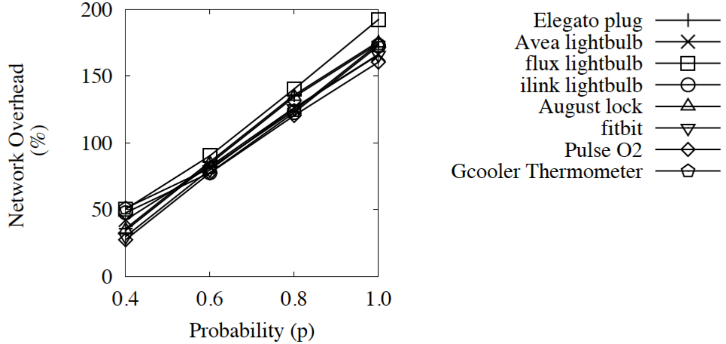


Fig. 11: The network bandwidth overhead of Max_Pad_Delay scheme.

Overhead of the Framework

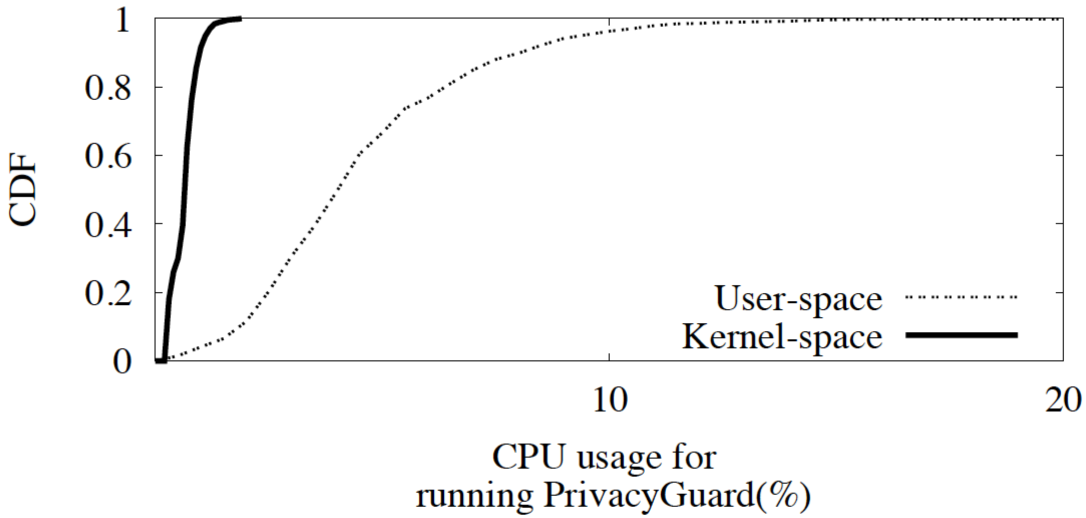


Fig. 12: Kernel-space and User-space CPU usage distribution of running PrivacyGuard.

Tamer Nadeem
tnadeem@vcu.edu
https://egr.vcu.edu/directory/tamer.nadeem/
https://music.lab.vcu.edu/