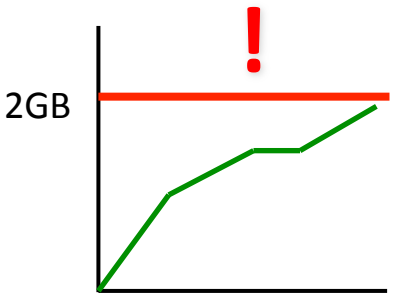
From Feast to Famine

Managing mobile network resources across environments and preferences

Rob Kiefer
Erik Nordström
Michael Freedman
Princeton University

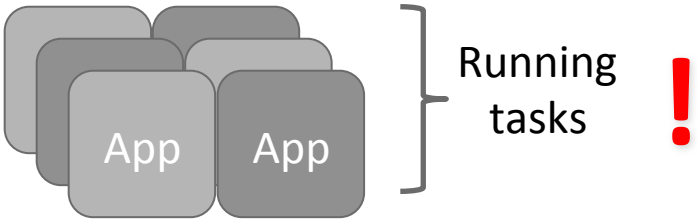
Network Usage Makes Demands on Limited Resources

- Data



- Battery

- Performance



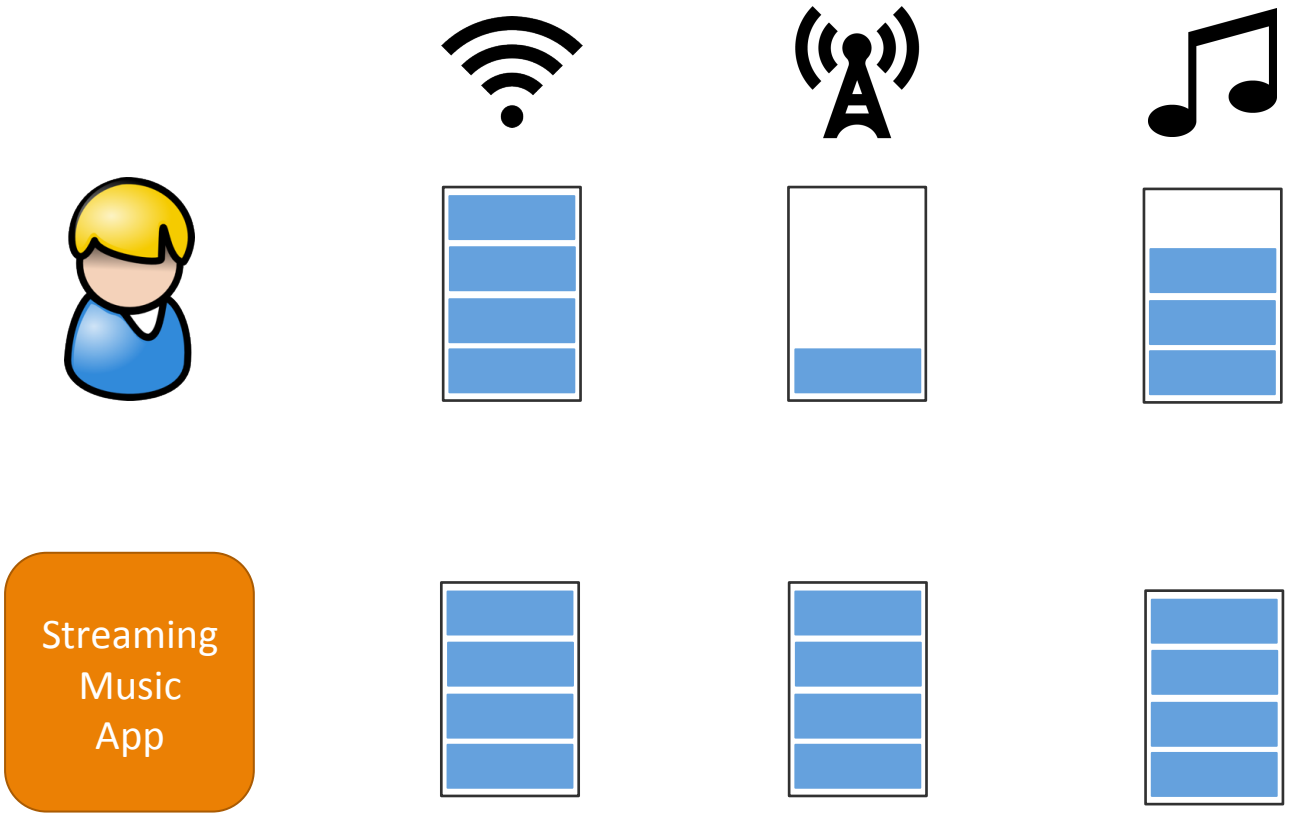
Limitations and Operating Conditions Are Not Static

- User mobility
  - indoor vs outdoor
  - WiFi vs LTE
  - multipath & migration
- User interest
  - foreground vs background apps
  - Interactive vs streaming

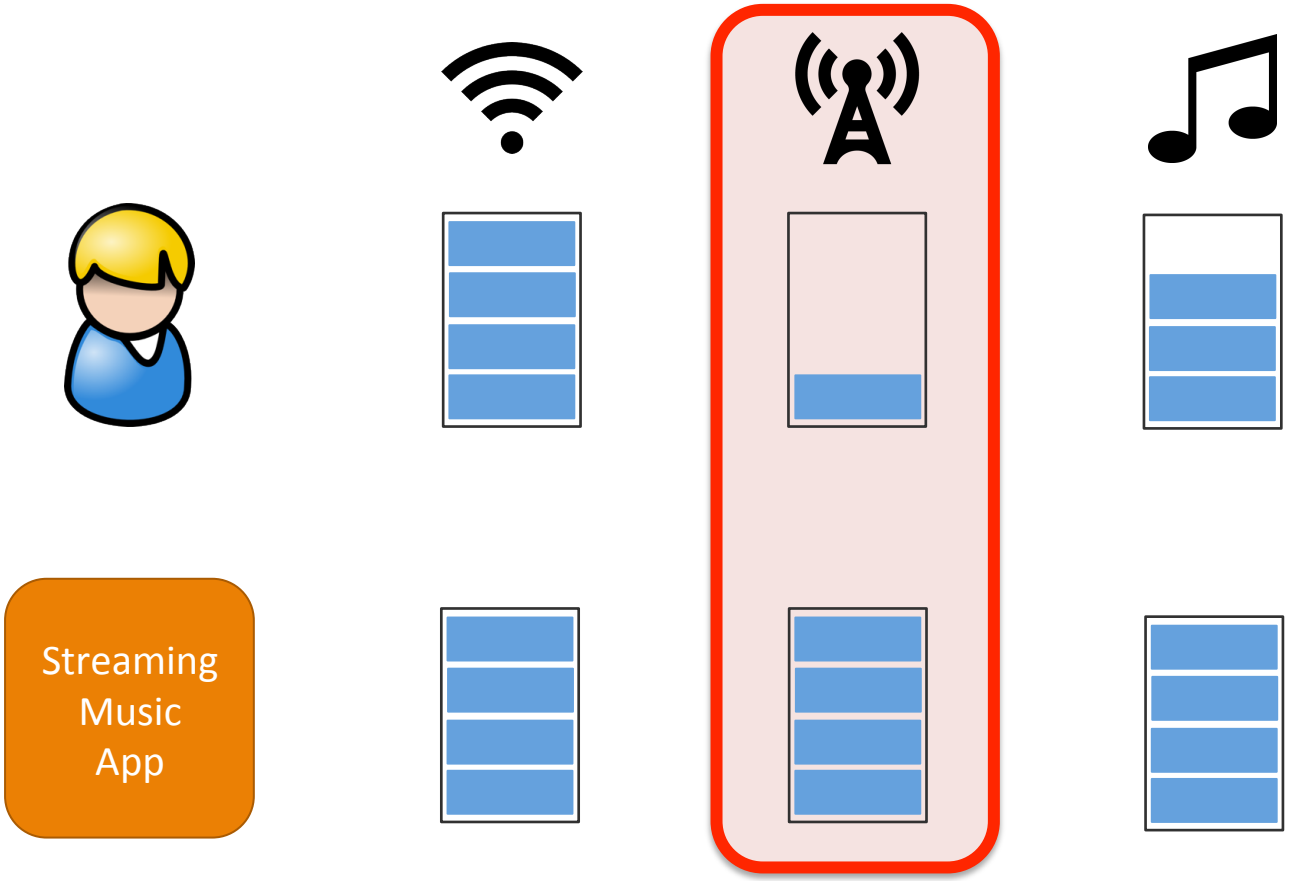
Divergent Goals = Resource Conflicts

- User and apps may differ
  - e.g., apps may prioritize perf, user is cost sensitive
- User has to moderate resource usage between apps

User Goals Don’t Match App Goals

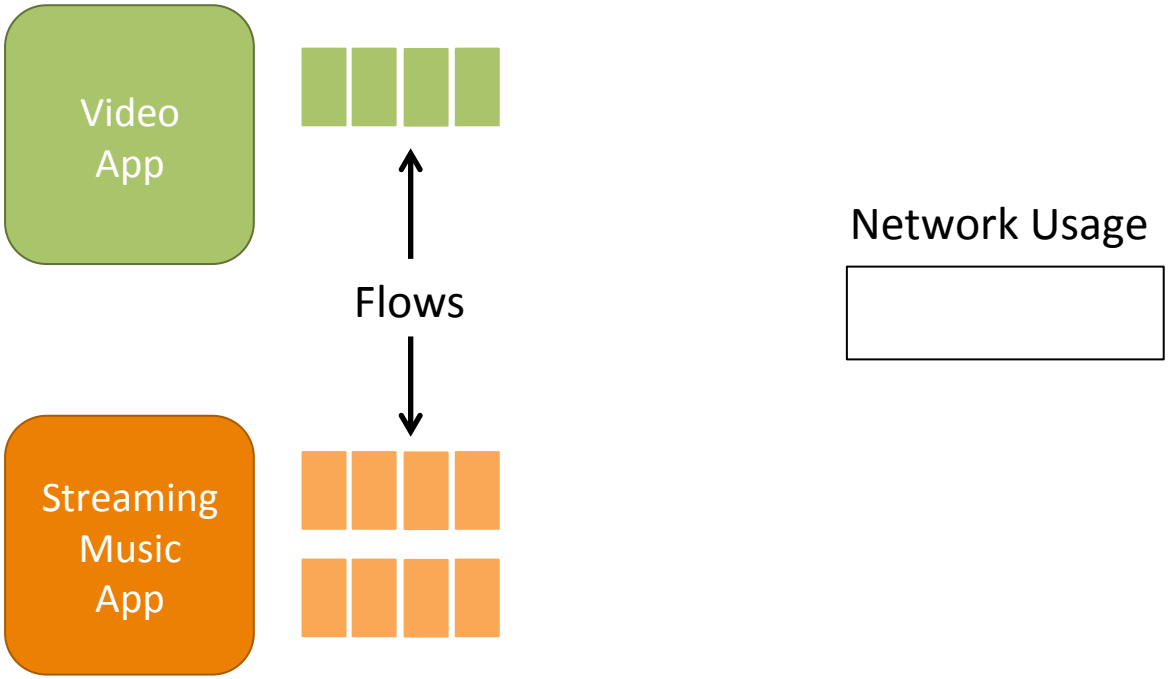


User Goals Don't Match App Goals

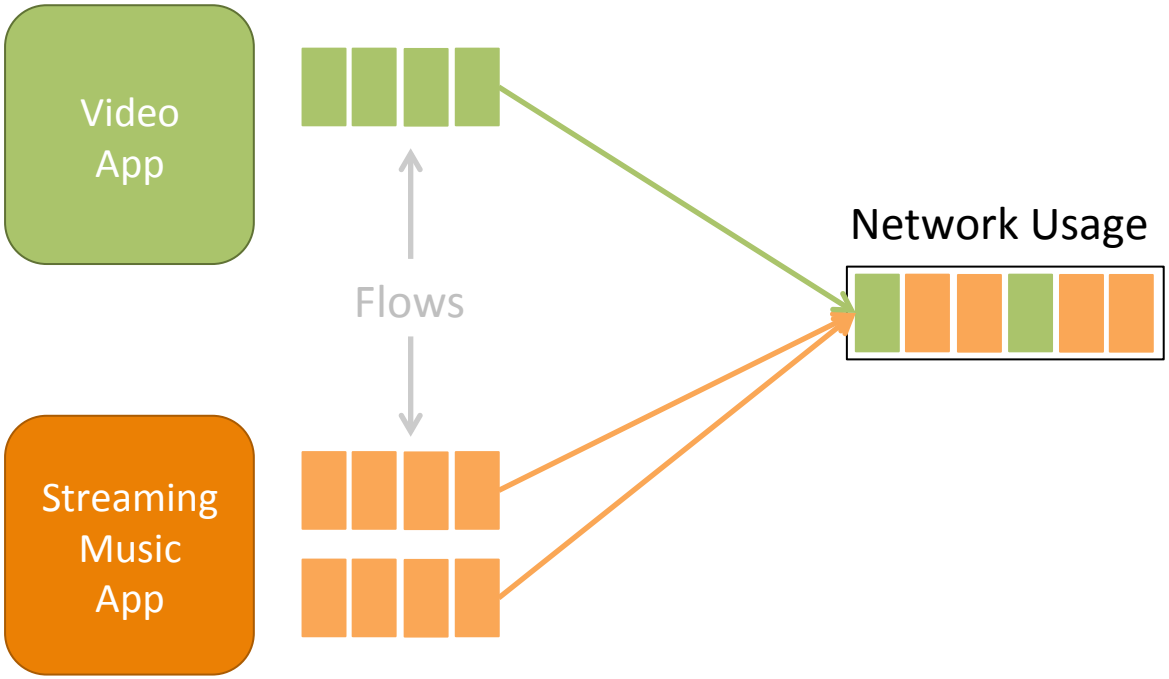


User wants to sacrifice quality; app overuses cell network

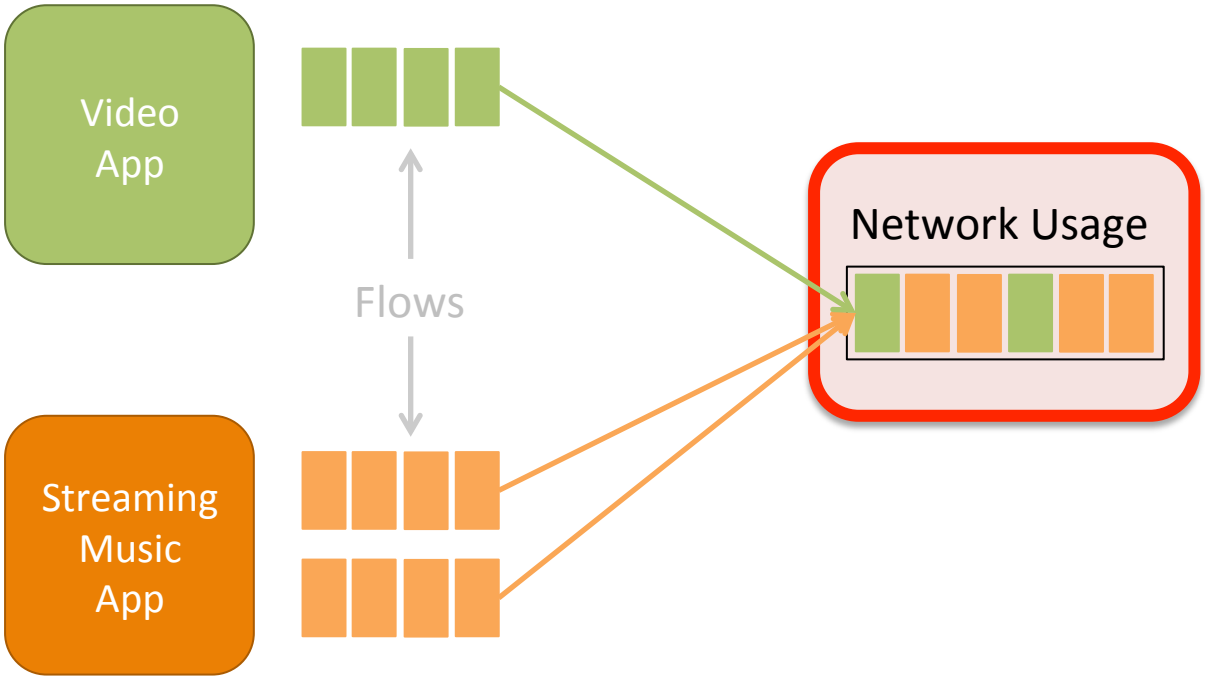
Flow-level Fairness != App-level Fairness



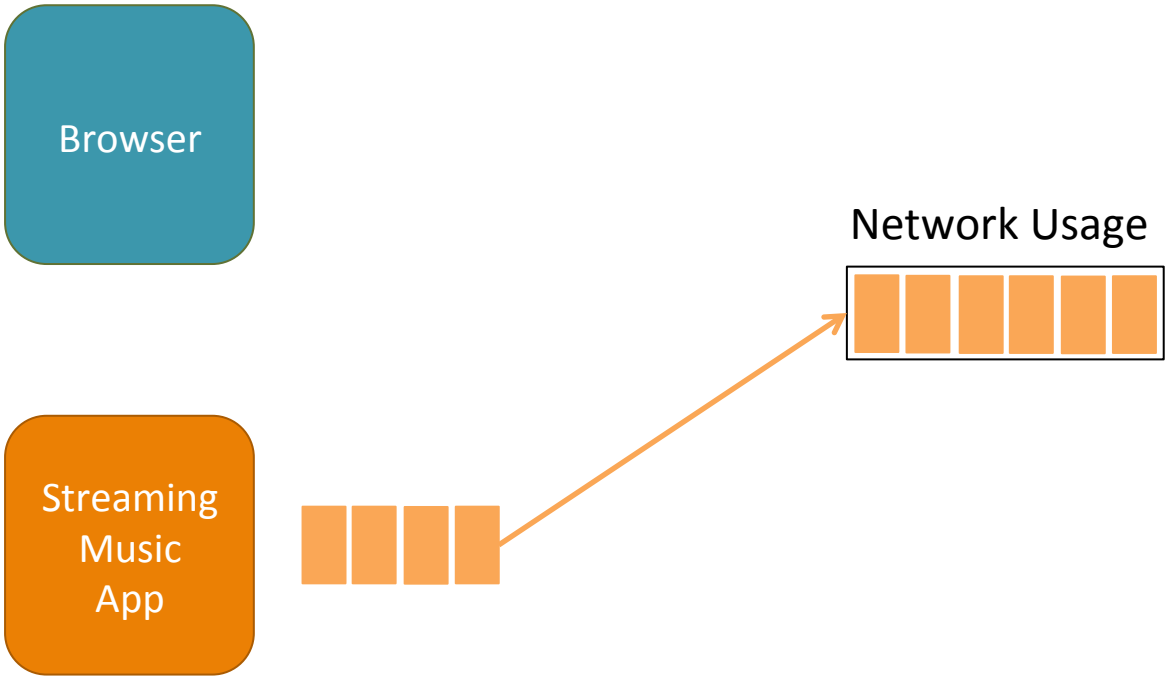
Flow-level Fairness != App-level Fairness



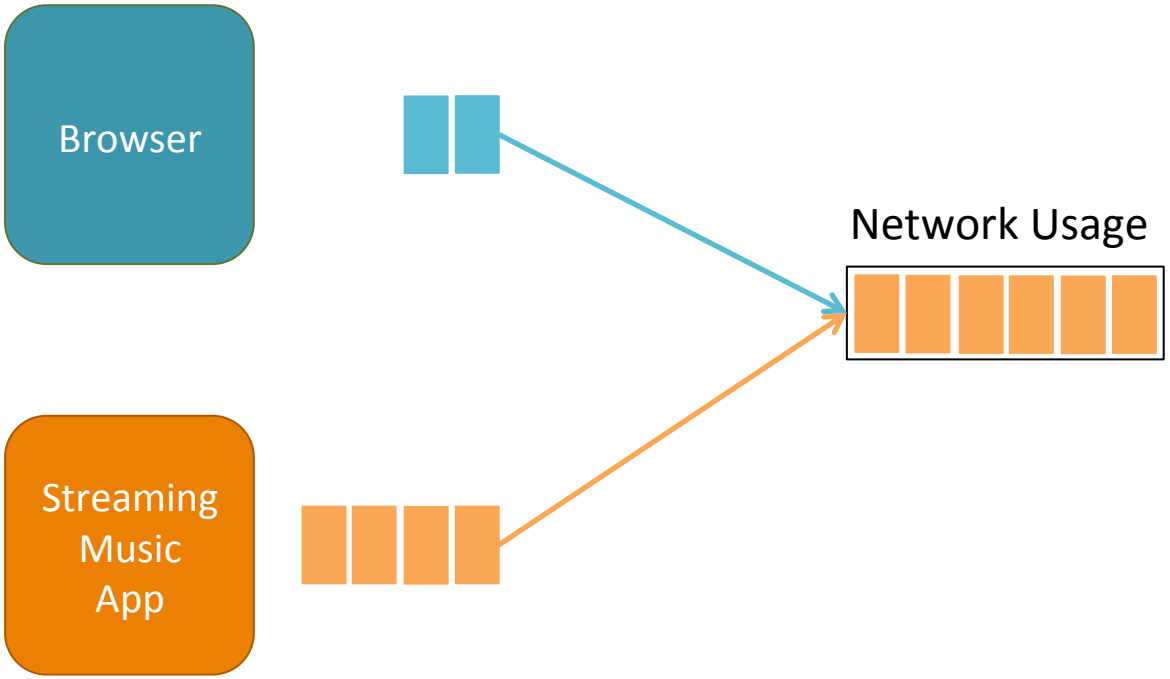
Flow-level Fairness != App-level Fairness



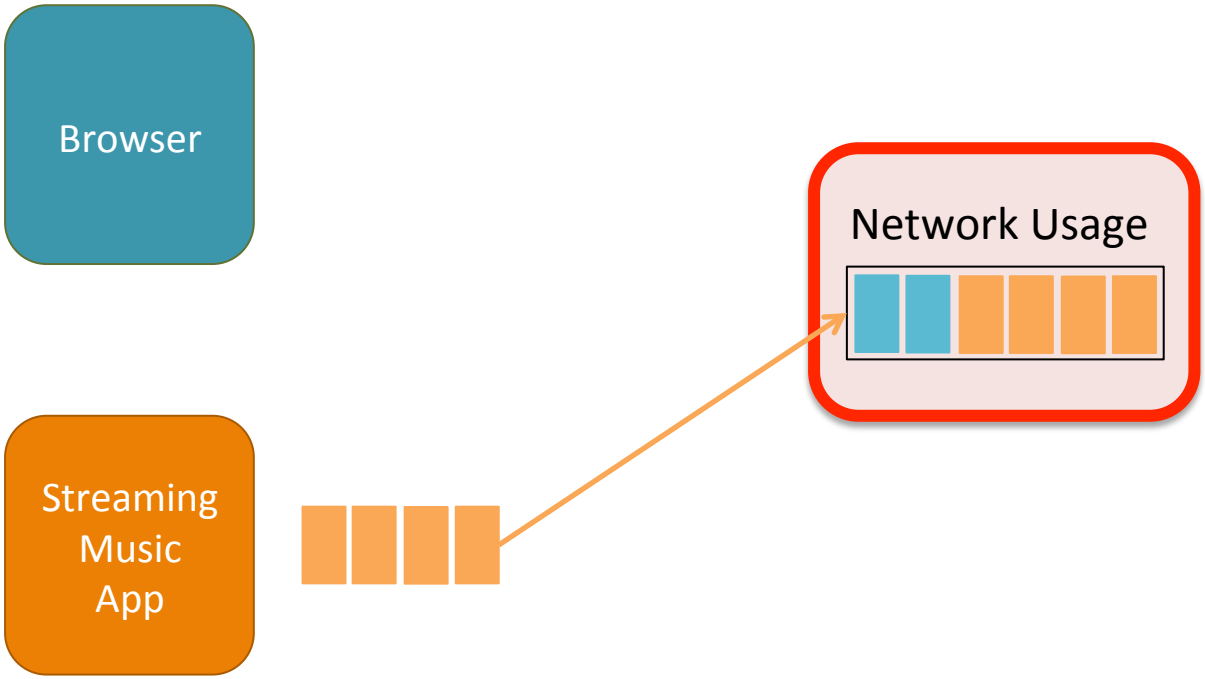
Interactive Apps Lack Prioritization



Interactive Apps Lack Prioritization

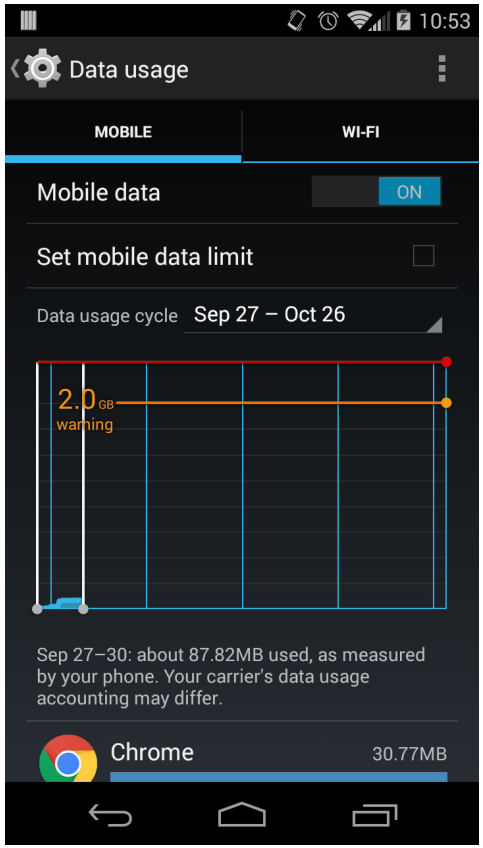


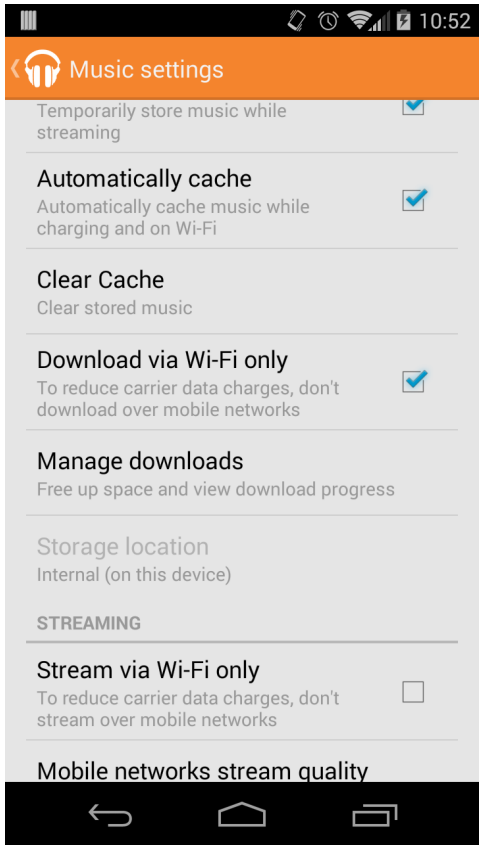
Interactive Apps Lack Prioritization

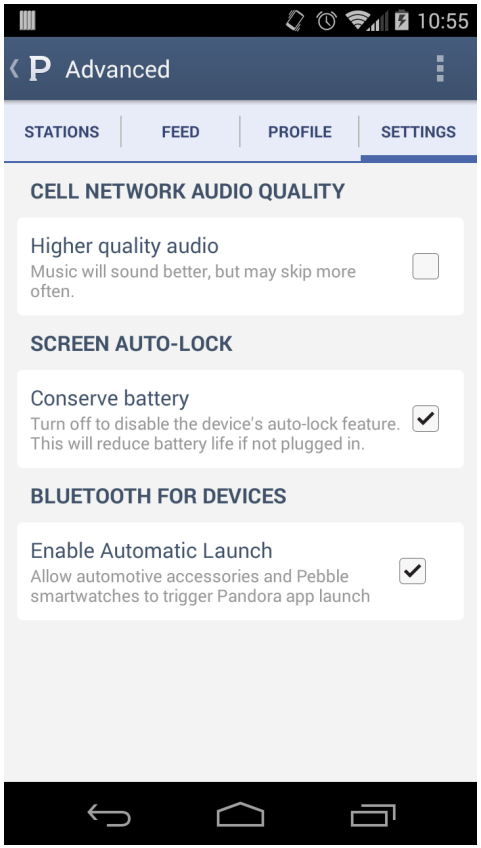


Interactive apps stuck behind long-running flows, hurting UX.

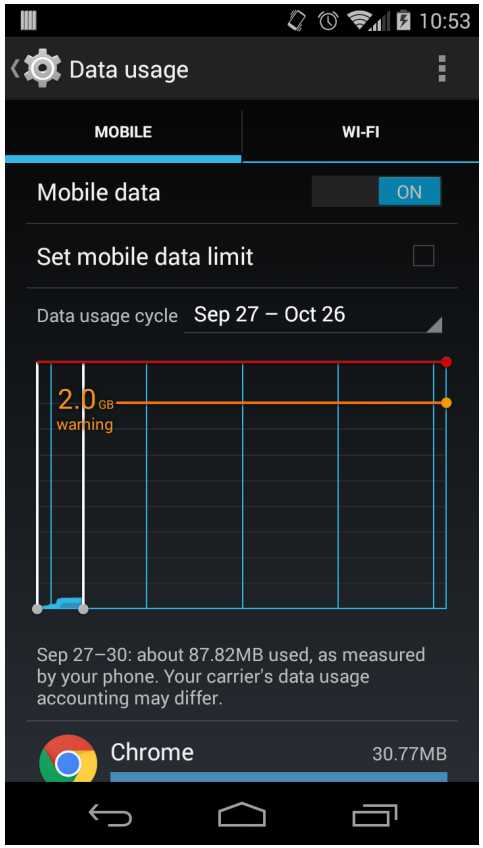
Strawman: Users micromanage across settings







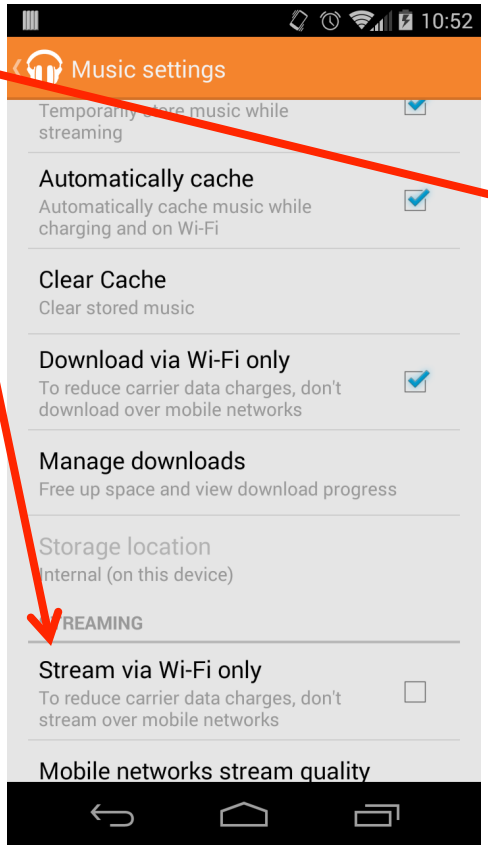
Strawman: Users micromanage across settings

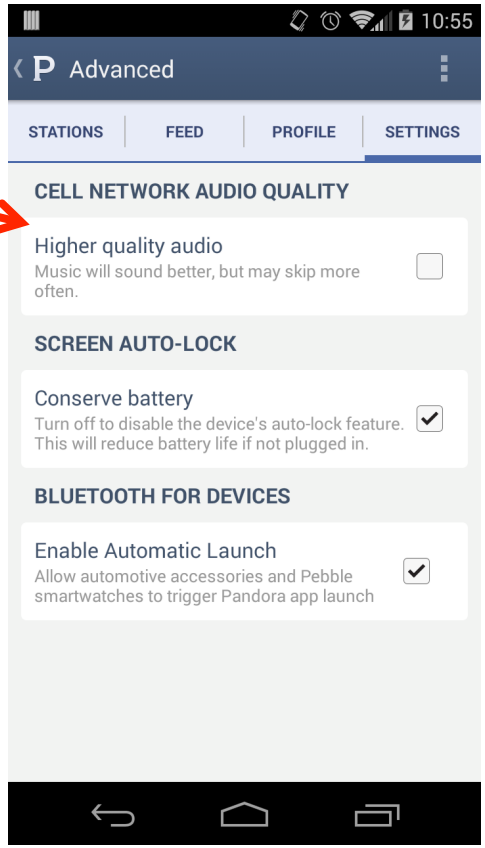


- Requires user to monitor and estimate usage over potentially long epochs (hours, days)
- Restricts typically all-or-nothing, e.g., allow on mobile or not

Strawman: Users micromanage across settings

Poor reuse in similar apps



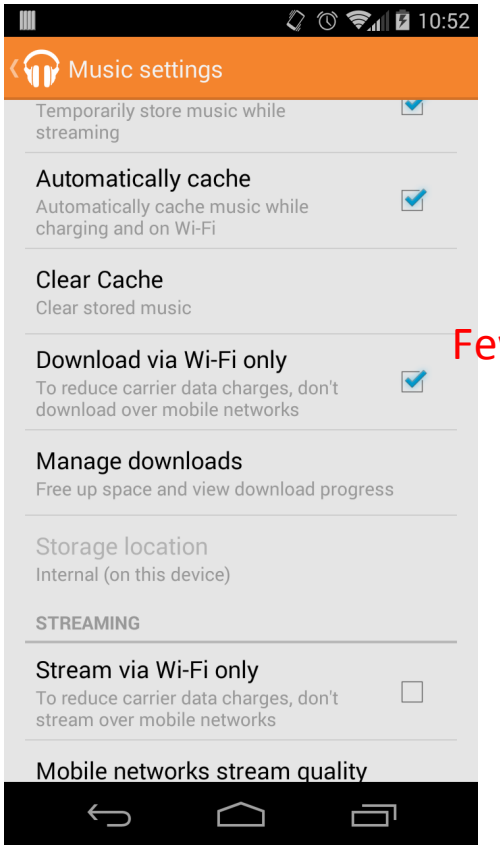


Strawman: Users micromanage across settings

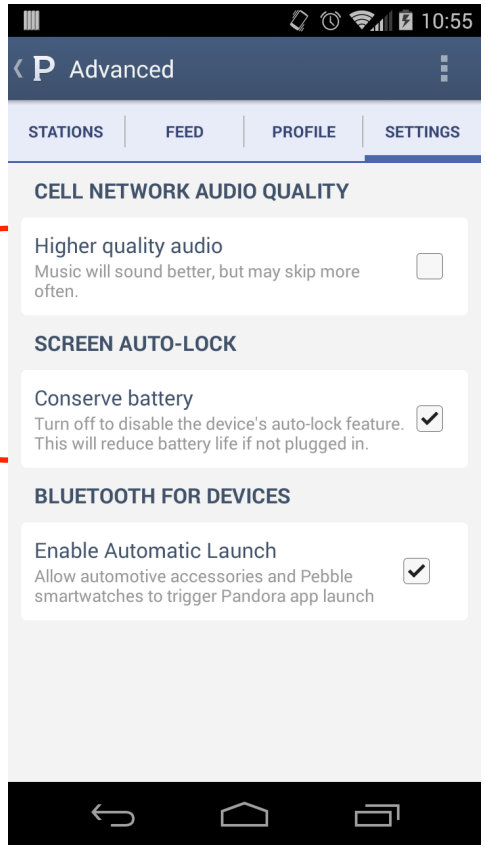
Poor reuse in similar apps

Varying levels of control

Many



Few

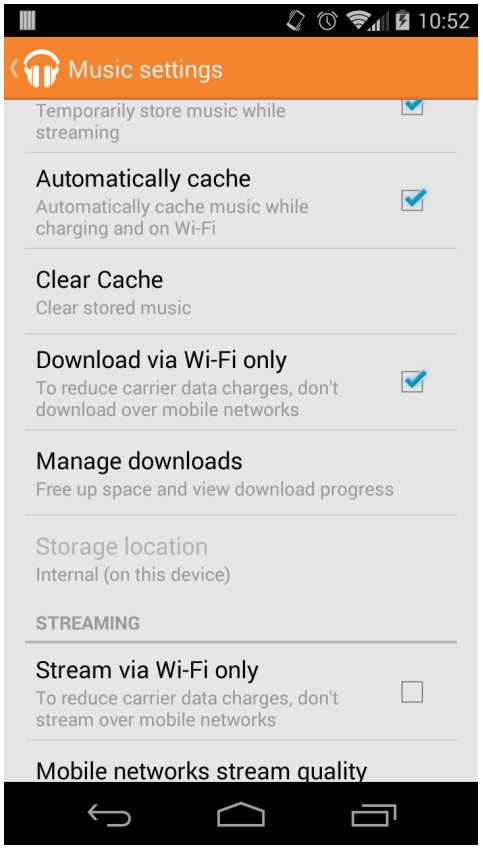


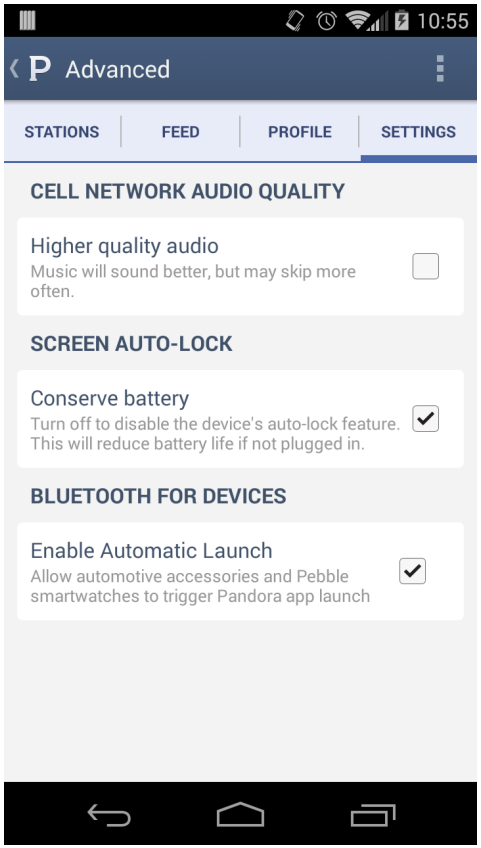
Strawman: Users micromanage across settings

Poor reuse in similar apps

Varying levels of control

Updates can add, remove, or change settings without user knowing





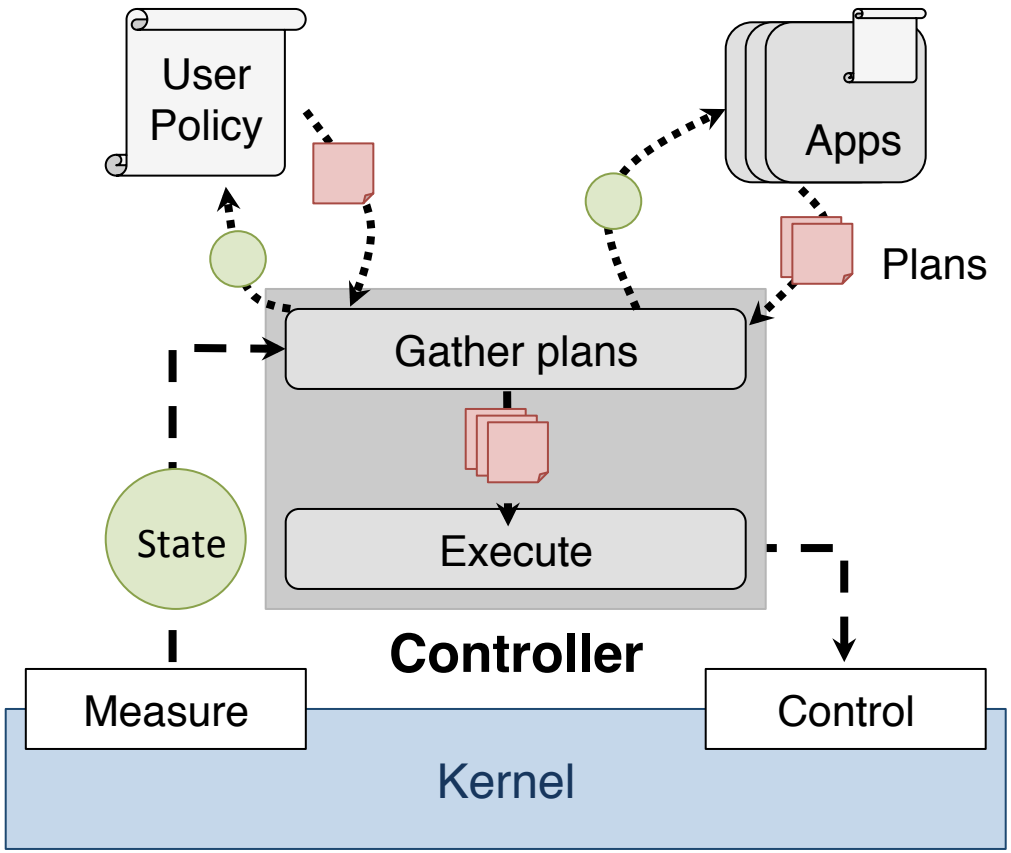
Strawman: OS-level limitations and restrictions

- OS limits background activity
- Only certain classes of apps
- Used by iOS and WinPhone
- User has no choice

Our Solution: Tango

- Usage preferences through programmatic policy; user is priority, but app flexibility
- Address conflicts *proactively*
- Migration / multipath in mind
- Handle dynamic operating conditions

Tango Overview

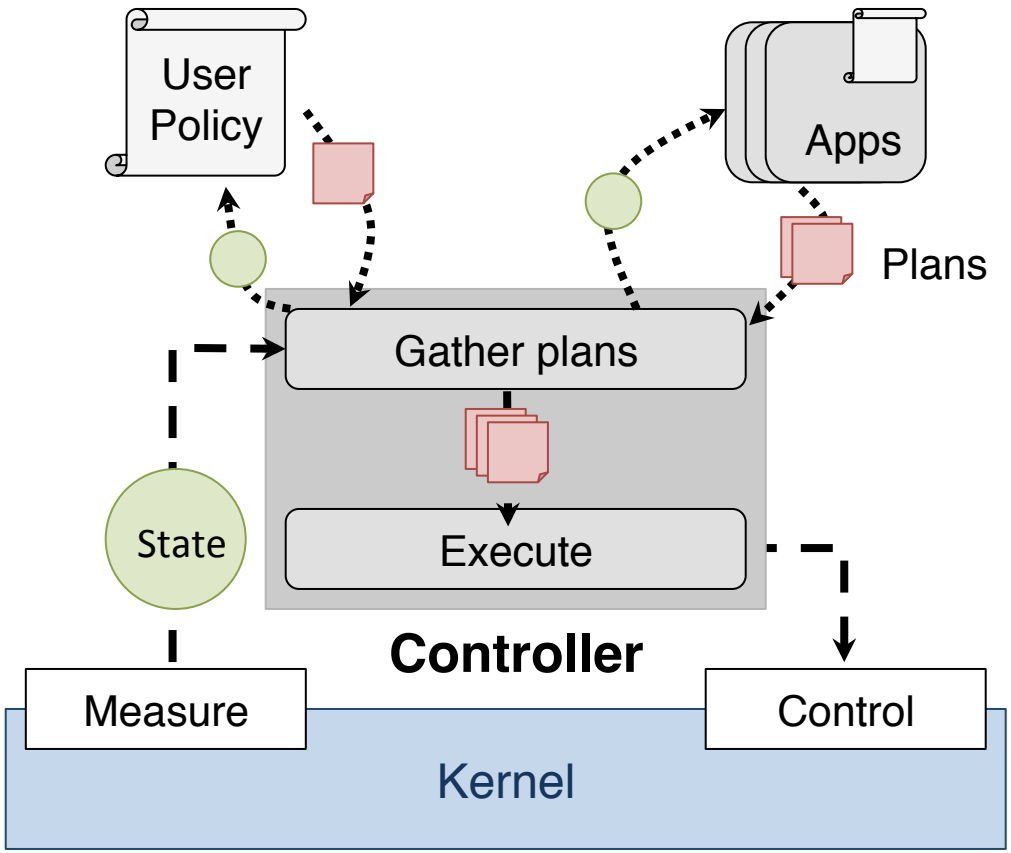


Tango Overview

State – metrics from kernel sources

Plans – actions to be taken

Policy – generates plans from given state

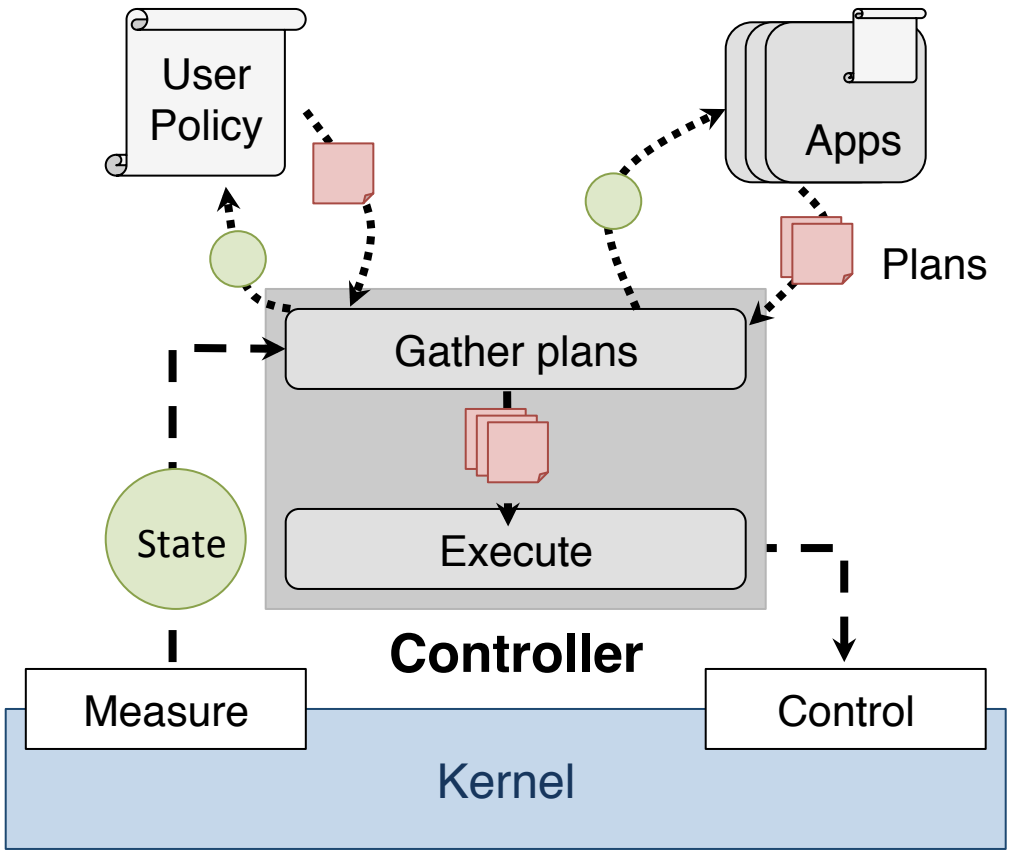


Tango Overview

1. Controller

2. User Policy

3. App Policies

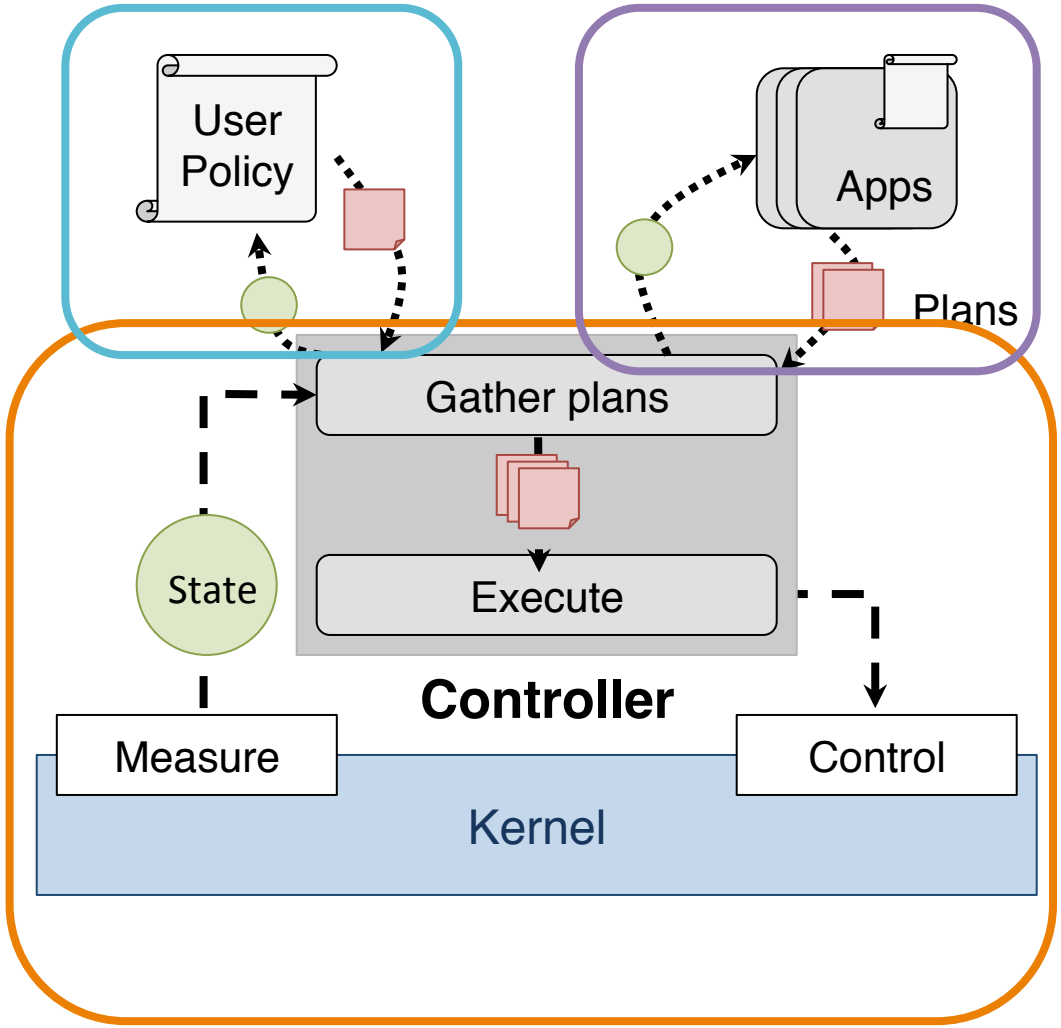


Tango Overview

1. Controller

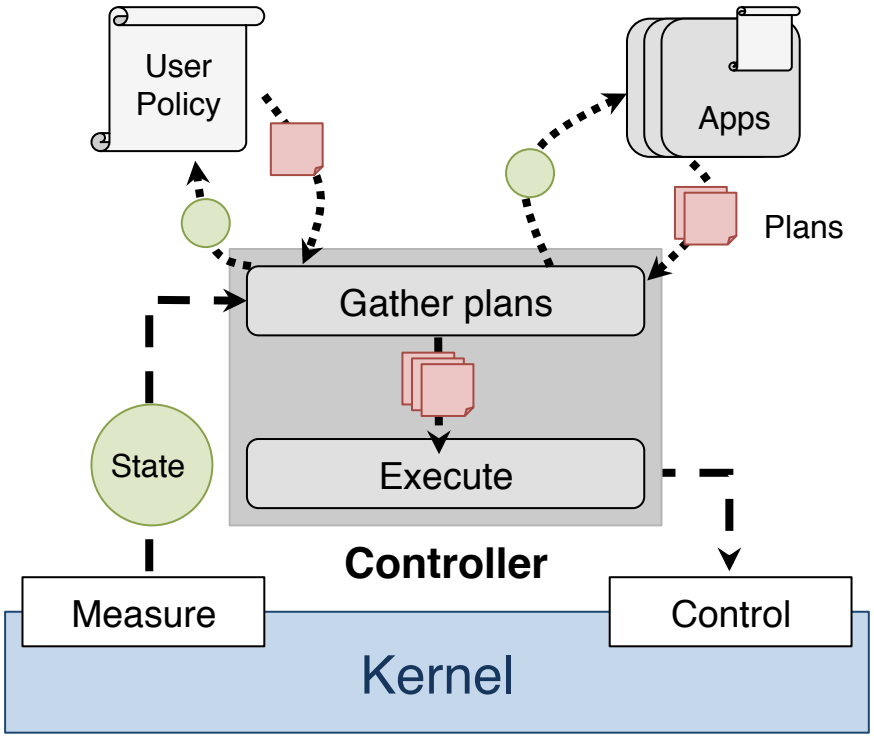
2. User Policy

3. App Policies



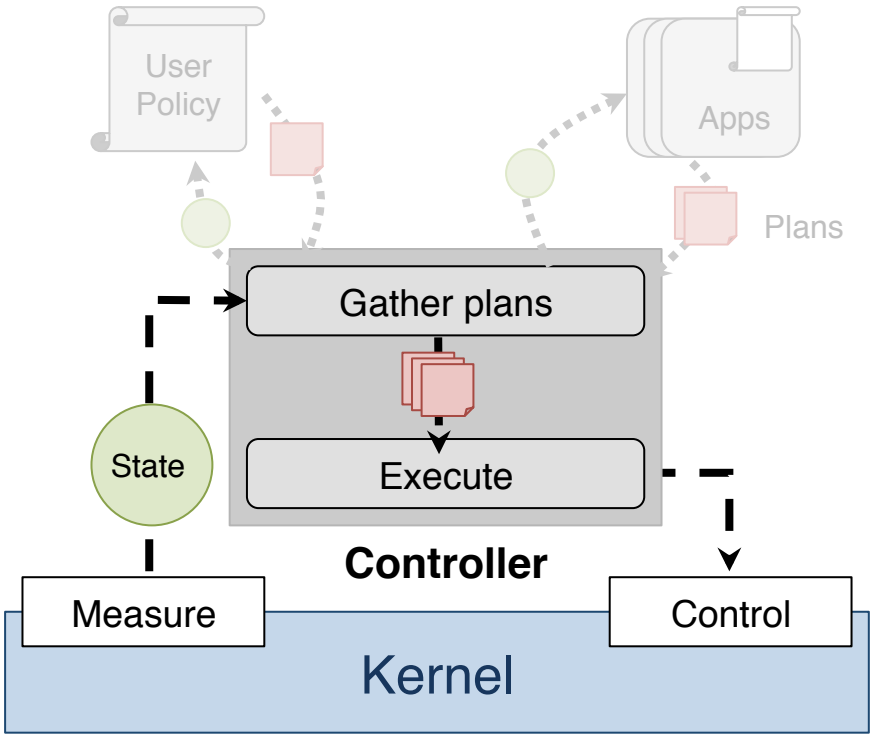
Tango: Controller

- Measures device **state**
- Generates policy constraints
- Carries out policy **plans**
  - Verify against constraints
  - Perform **actions** in plan



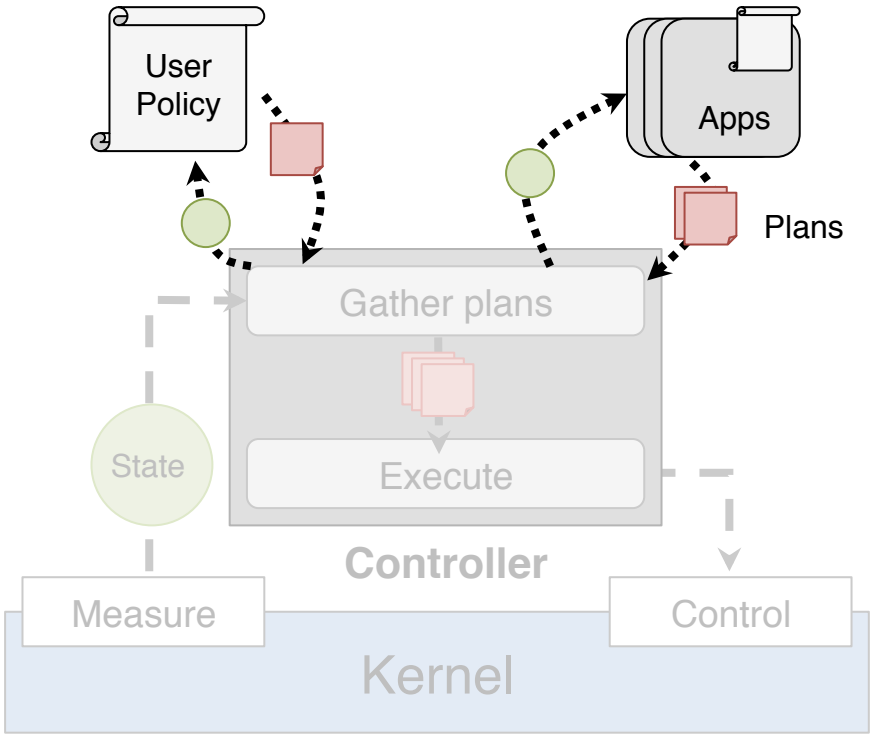
Tango: Controller

- Measures device **state**
- Executes the user policy
- Carries out policy **plans**
  - Verify against constraints
  - Perform **actions** in plan



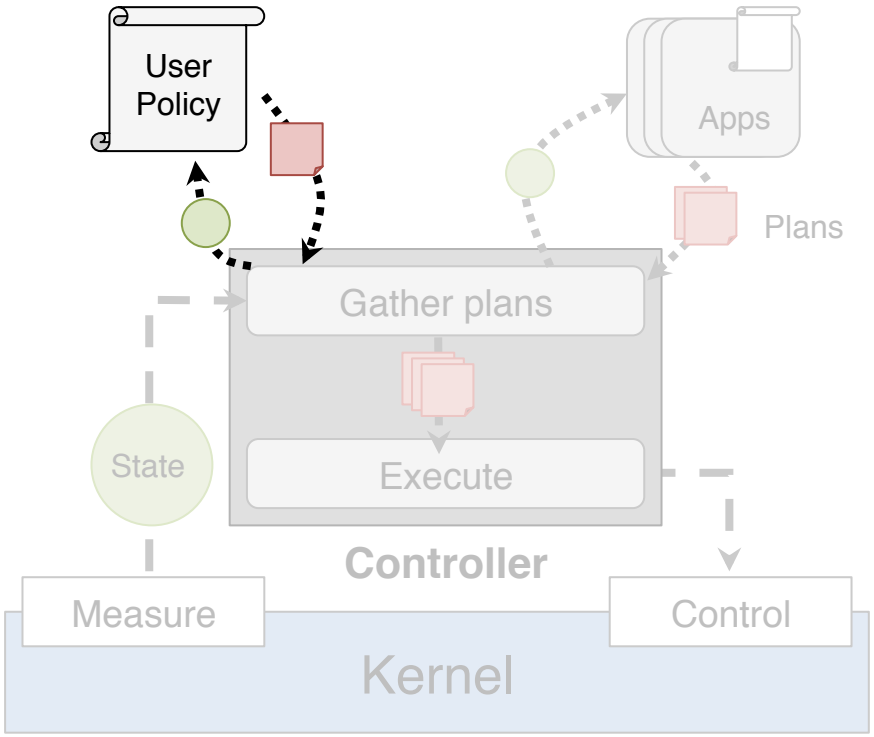
Tango: Policies

- Programs
- Turn state into plans
- Two levels: user & app



Tango: Policies

- Programs
- Turn state into plans
- Two levels: user & app



- User level
  - Global management of network resources
  - Sets resource constraints for app policies
  - Default plan for (classes of) apps

Policies: Turning State into Actions

Policies: Turning State into Actions

net_name: eth0
net_type: CELL,
batt_charge: false
batt_perc: 50
flows: 10, 11, 12…
…

Policies: Turning State into Actions

State

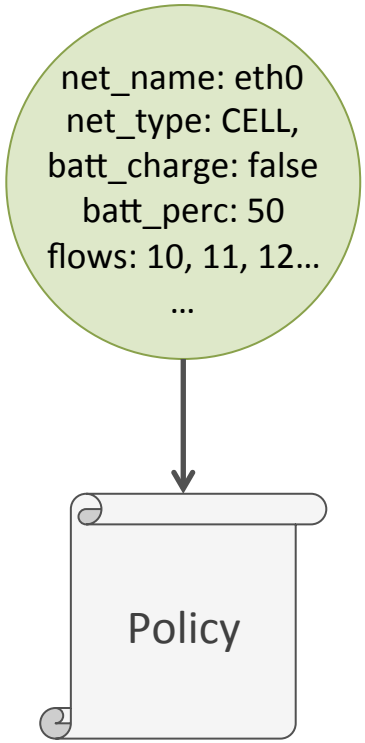
Source	Info
Transport Layer	# retrans., RTTs, cong. window…
Network Layer (IP)	addresses, routing rules…
Link Layer	type, signal quality, bit errors…
Battery	charging status, percent…

net_name: eth0
net_type: CELL,
batt_charge: false
batt_perc: 50
flows: 10, 11, 12…
…

Policies: Turning State into Actions

State

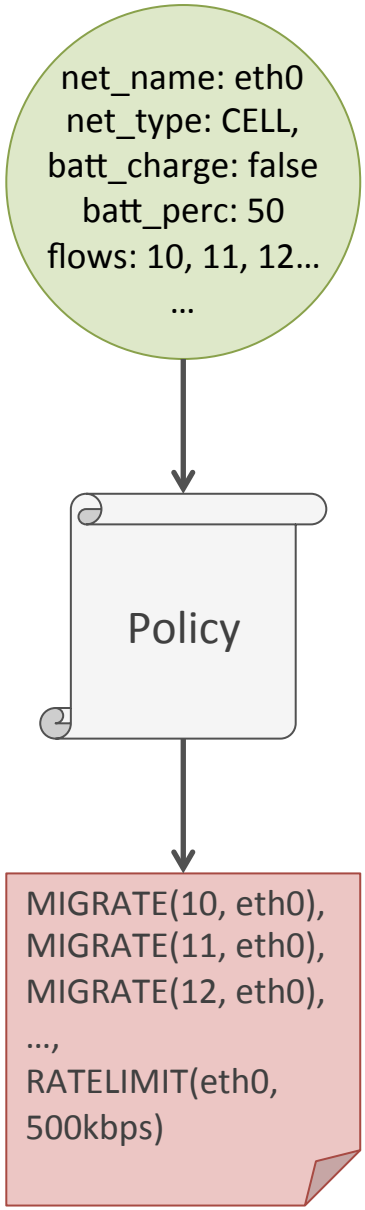
Source	Info
Transport Layer	# retrans., RTTs, cong. window…
Network Layer (IP)	addresses, routing rules…
Link Layer	type, signal quality, bit errors…
Battery	charging status, percent…



Policies: Turning State into Actions

State

Source	Info
Transport Layer	# retrans., RTTs, cong. window…
Network Layer (IP)	addresses, routing rules…
Link Layer	type, signal quality, bit errors…
Battery	charging status, percent…



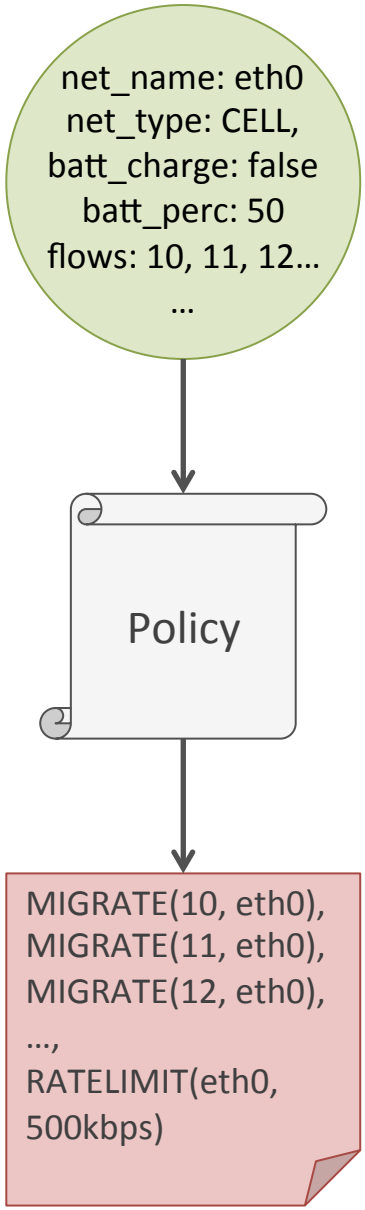
Policies: Turning State into Actions

State

Source	Info
Transport Layer	# retrans., RTTs, cong. window…
Network Layer (IP)	addresses, routing rules…
Link Layer	type, signal quality, bit errors…
Battery	charging status, percent…

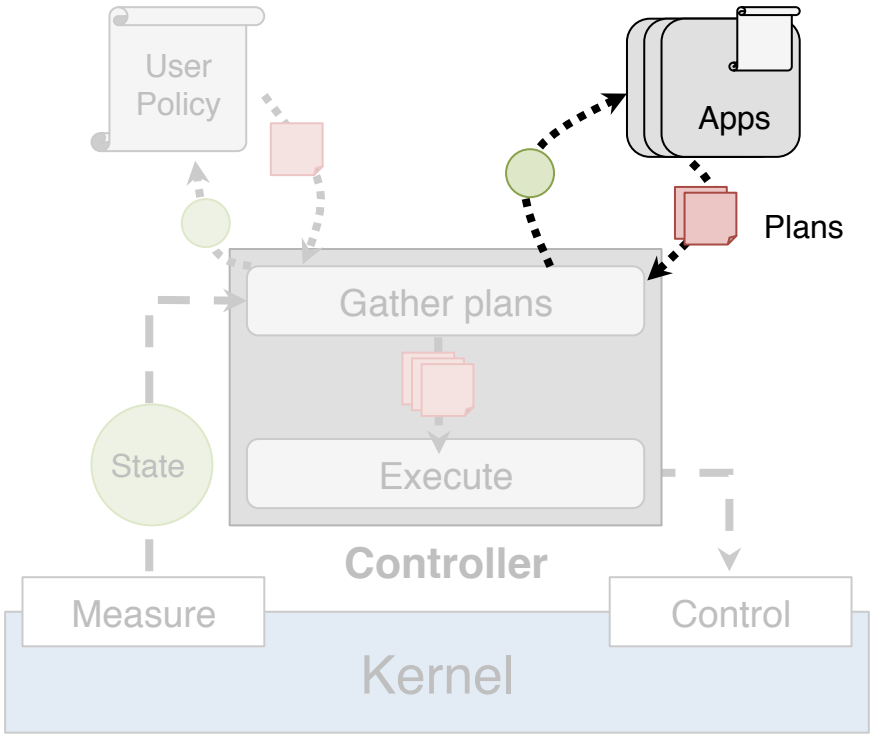
Actions

Action	Iface	Flow
ENABLE	✔	✔
RATELIMIT	✔	✔
LOG	✔	✔
MANAGE	✔	
MIGRATE		✔



Tango: App Policies

- Leverage *local* state
- Only act on their flows
- Subject to user policy **constraints**
- App can provide **hints**



Tango: Constraints & Hints

- Proactively address resource conflicts
  - Constraints known *a priori*
  - Simplifies controller & app policy interaction cycle
- Apps can hint at desired resource needs for future rounds

Tango: User & App Policy

USER POLICY

```
evaluate(s, c):
    // Plan returned.

genConstraints(s):
    c = new Constraints()
    for a in s.apps():

        if a.isForeground():
            c.put(a, HIGH)




    return c
```

Tango: User & App Policy

USER POLICY

```
evaluate(s, c):
    // Plan returned.

genConstraints(s):
    c = new Constraints()
    for a in s.apps():

        if a.isForeground():
            c.put(a, HIGH)



    return c
```

APP POLICY

```
evaluate(s, c_a):
    plan = new Plan()
    pstate = GetPlayerState()
    // Rest of policy omitted.

    plan.hintPriority = NORM
    urgent = false
    buffer = pstate.bufferTime
    if buffer > 30:
        urgent = false
    elif buffer < 20 || urgent:
        urgent = true
        plan.hintPriority = HIGH

    return plan
```

Tango: User & App Policy

USER POLICY

```
evaluate(s, c):
    // Plan returned.

genConstraints(s):
    c = new Constraints()
    for a in s.apps():
        hint = a.hintPriority
        if a.isForeground():
            c.put(a, HIGH)
        elif Allowed(a, hint):
            c.put(a, hint)

    return c
```

APP POLICY

```
evaluate(s, c_a):
    plan = new Plan()
    pstate = GetPlayerState()
    // Rest of policy omitted.

    plan.hintPriority = NORM
    urgent = false
    buffer = pstate.bufferTime
    if buffer > 30:
        urgent = false
    elif buffer < 20 || urgent:
        urgent = true
        plan.hintPriority = HIGH

    return plan
```

Prototype

- Controller, sample policies, sample apps in Java for Android phones
- Flow migration provided by Serval (NSDI '12)
- Explored the policy space for single and multiple apps

Evaluation: Streaming Music Across Spotty Campus WiFi

- Campus WiFi offers chance to offload
- But, even seamless switching w/ migration has problems

Evaluation: Streaming Music Across Spotty Campus WiFi

- Campus WiFi offers chance to offload
- But, even seamless switching w/ migration has problems
- How to use policy to improve when to switch?
- How to minimize cell usage?
- How can app policies help?

Clinging to WiFi Leads to Poor Performance

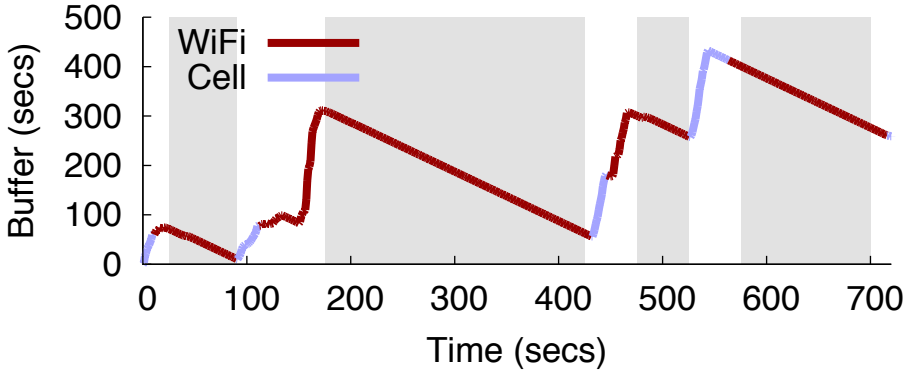
Good WiFi

Bad WiFi

Clinging to WiFi Leads to Poor Performance

Good WiFi

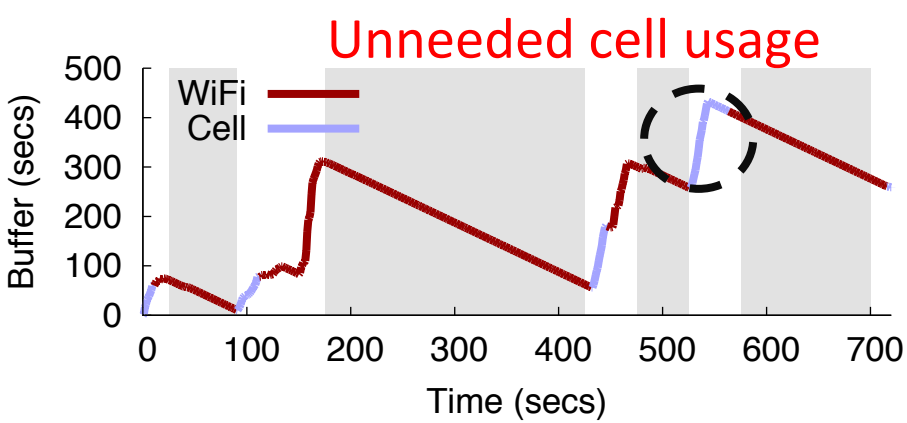
Bad WiFi



Clinging to WiFi Leads to Poor Performance

Good WiFi

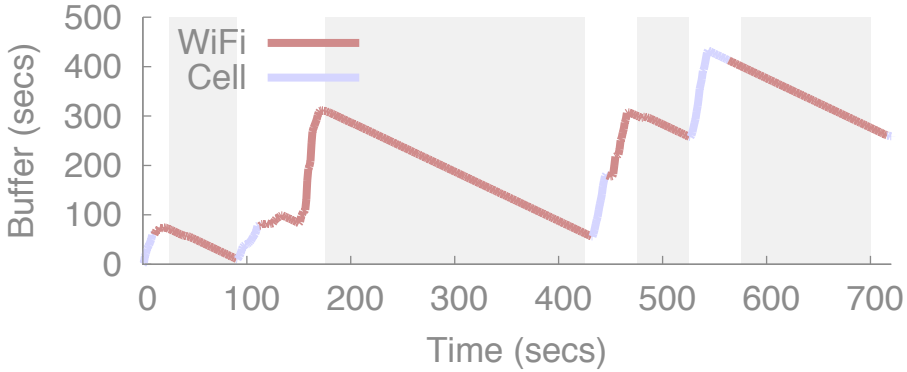
Bad WiFi

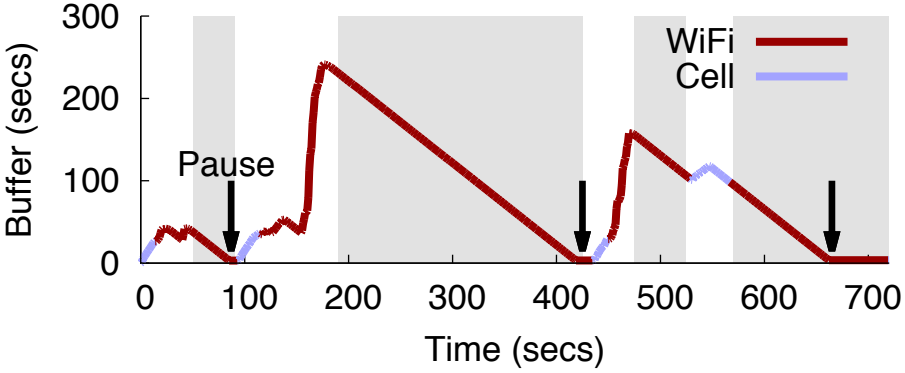


Clinging to WiFi Leads to Poor Performance

Good WiFi

Bad WiFi

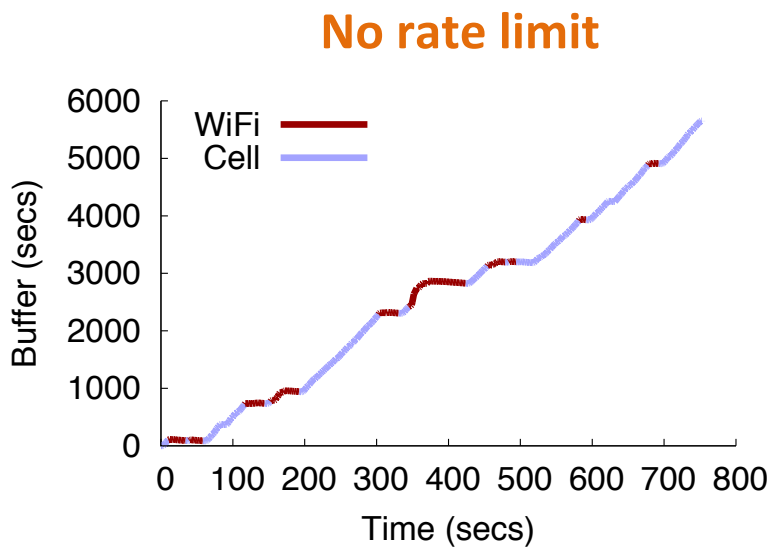




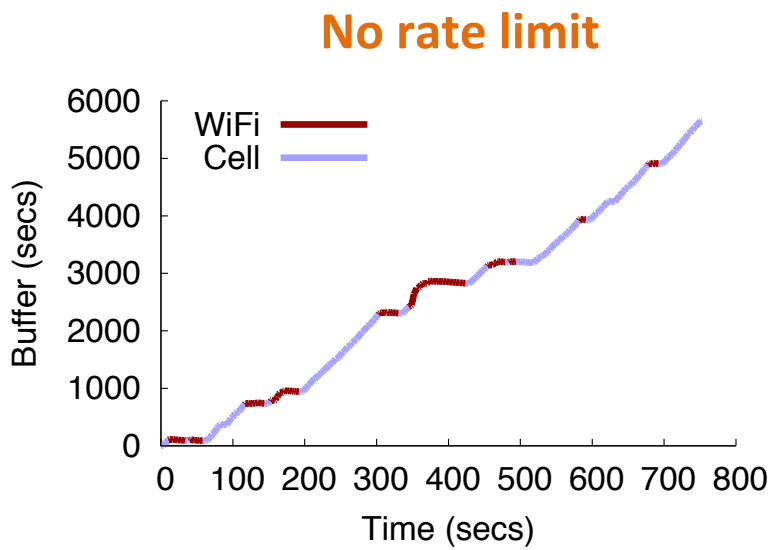
Improving Network Switching

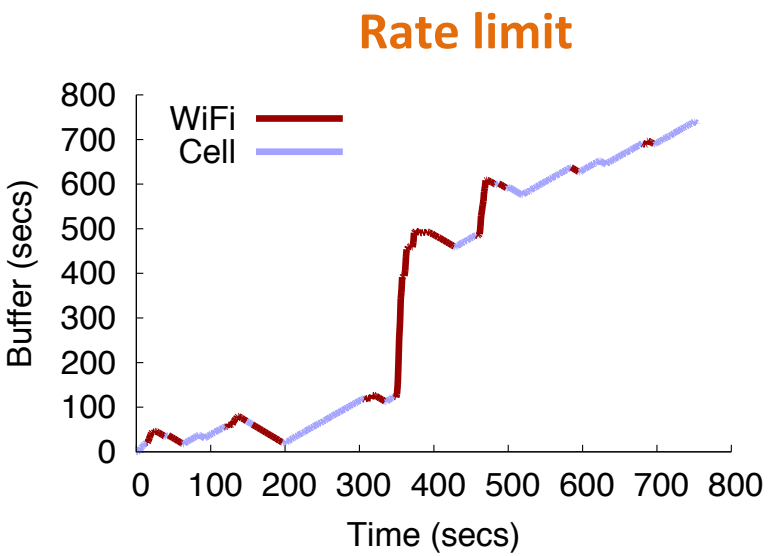
- More aggressive in leaving WiFi; more conservative in joining
- Use heuristics on signal strength trends
- Based on measurements across campus of signal strength vs achieved throughput

Improving Streaming Music Through Policy



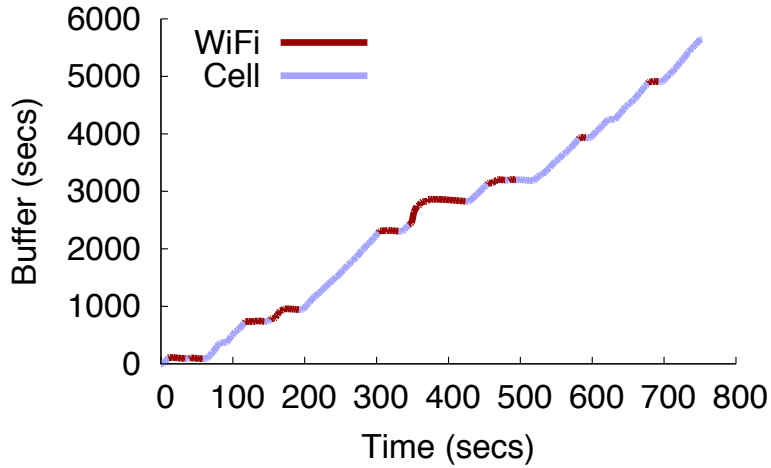
Improving Streaming Music Through Policy



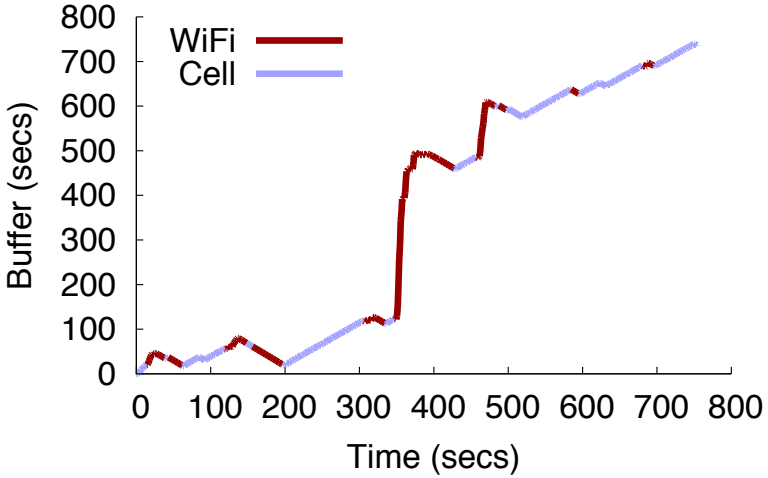


Improving Streaming Music Through Policy

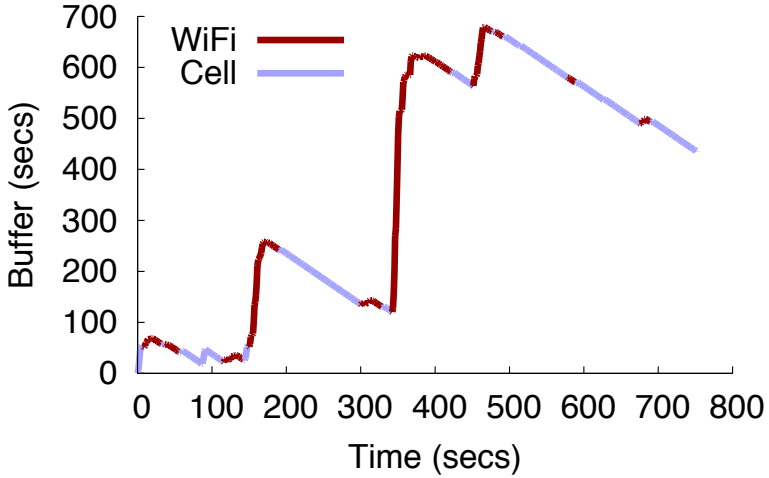
No rate limit



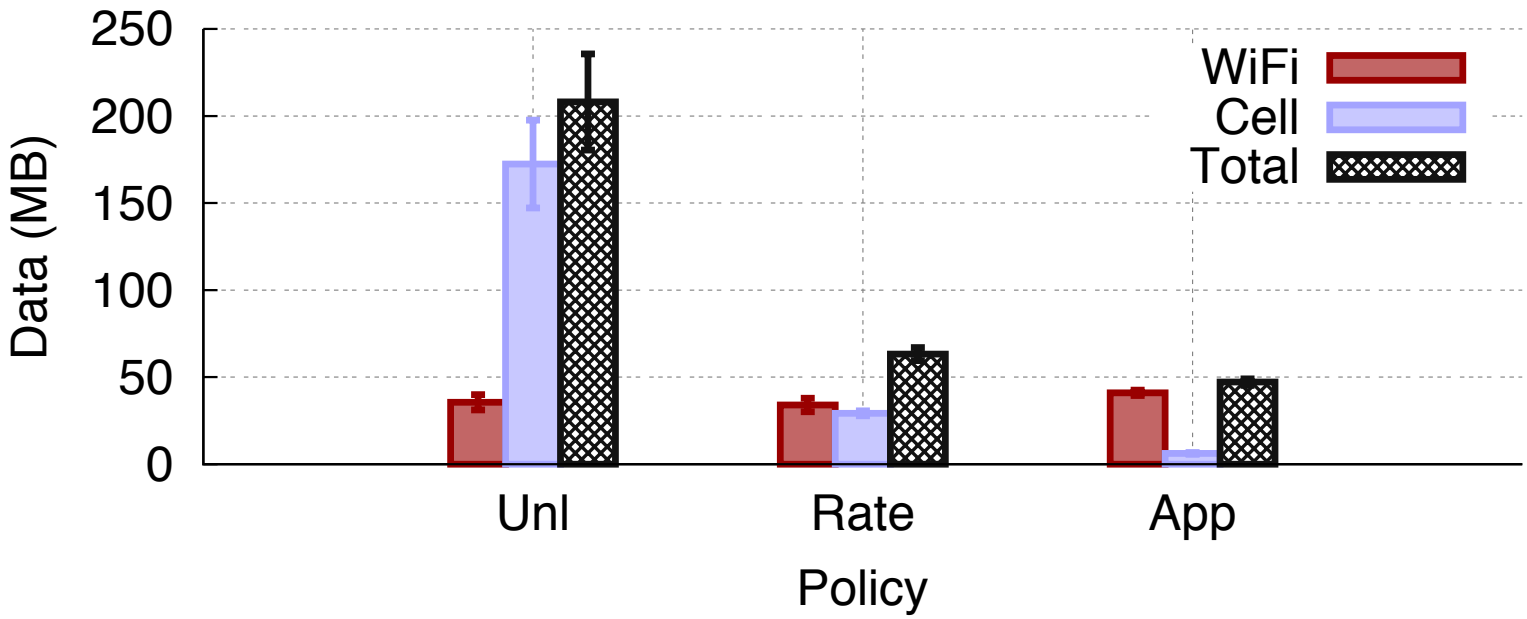
Rate limit



App Policy



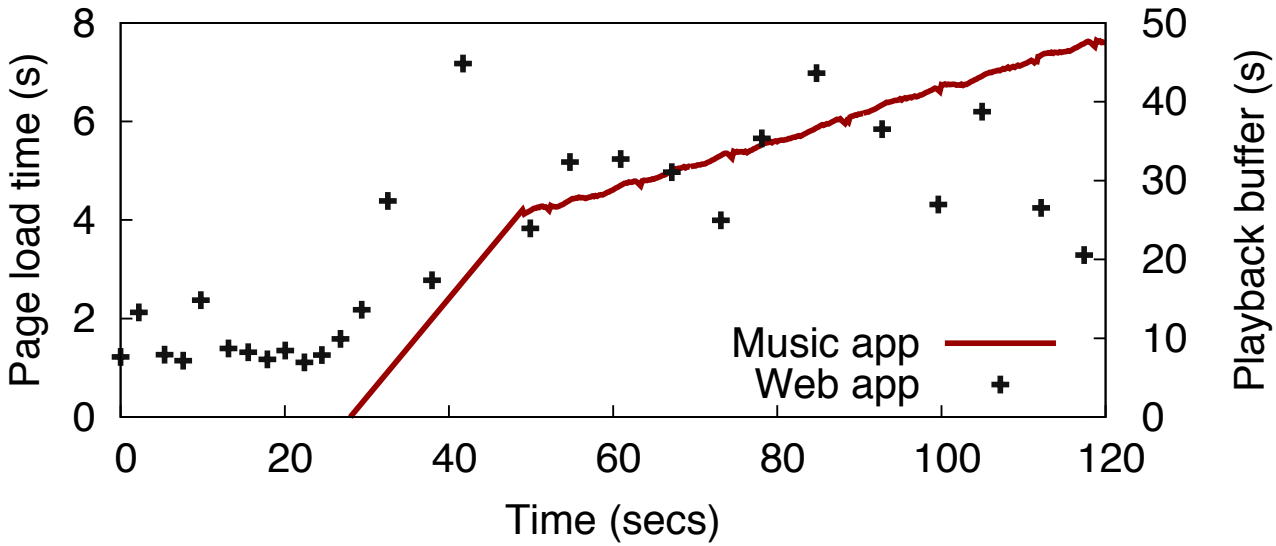
User and App Policy Cooperation Yields Best Usage



Rate uses 5-6x less cell data than Unl.

App uses about 30x less cell data than Unl.

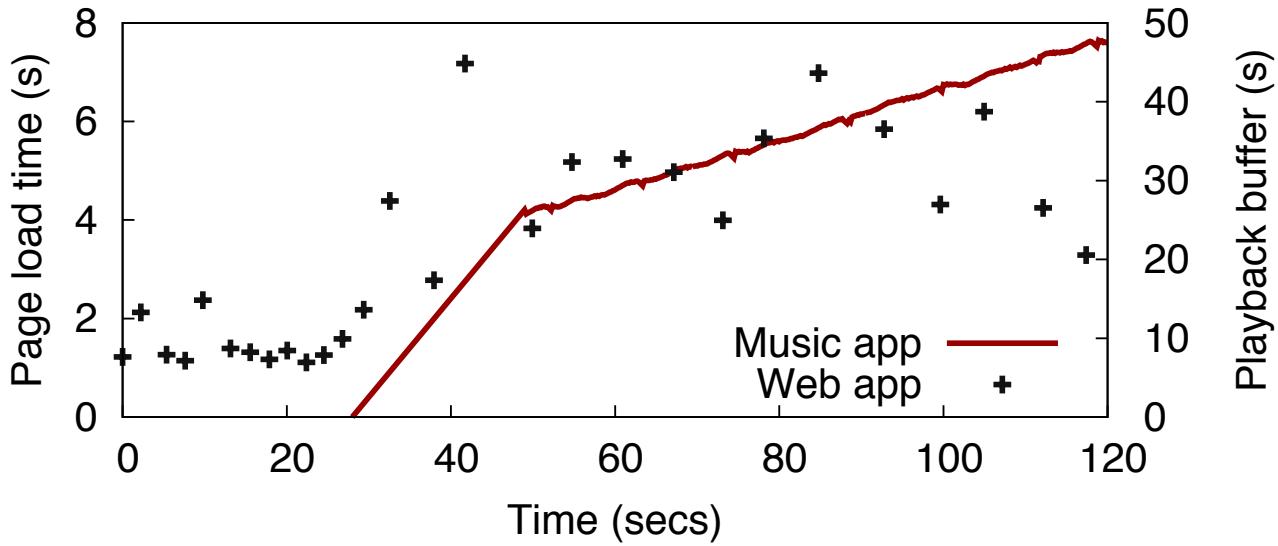
Interactive App Competing w/ Streaming App



Problem

Poor resource isolation leads to erratic load times

Interactive App Competing w/ Streaming App



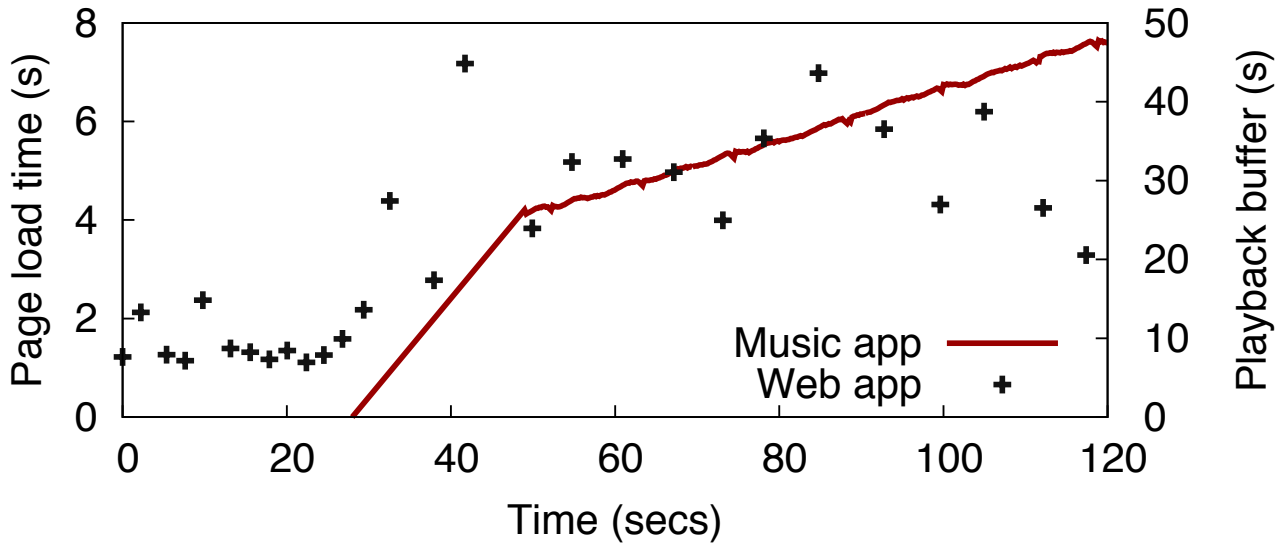
Problem

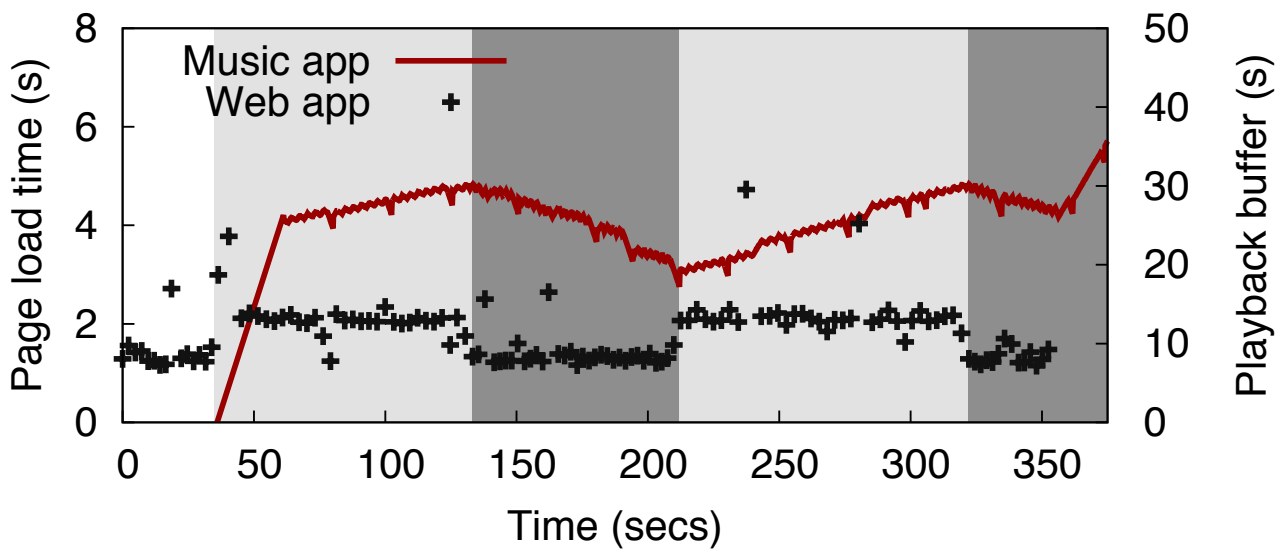
Poor resource isolation leads to erratic load times

Solution

App fair share, prioritize music only when buffer low

Interactive App Competing w/ Streaming App





Related Work

- Context awareness: CASS, CARISMA, JCAF
  - Using context to improve apps
  - Not looking at resource usage; not single device

Related Work

- Context awareness: CASS, CARISMA, JCAF
  - Using context to improve apps
  - Not looking at resource usage; not single device

- Network Choice/WiFi Offloading
  - Many: Whiffler, IMP, BreadCrumbs, SALSA…
  - Complementary to Tango

Related Work

- Context awareness: CASS, CARISMA, JCAF
  - Using context to improve apps
  - Not looking at resource usage; not single device

- Network Choice/WiFi Offloading
  - Many: Whiffler, IMP, BreadCrumbs, SALSA…
  - Complementary to Tango

- Serval, MPTCP, TCP-Migrate
  - Need policy to guide decisions on nets to use
  - Complementary to Tango

Conclusions

- Tango is a network resource management framework based on programmatic policy
- Proactively handle resource conflicts while including user and app prefs into decisions
- Demonstrated the value it adds for single apps as well as across apps

Thanks! Questions?