Selectively Taming Background Android Apps to Improve Battery Lifetime



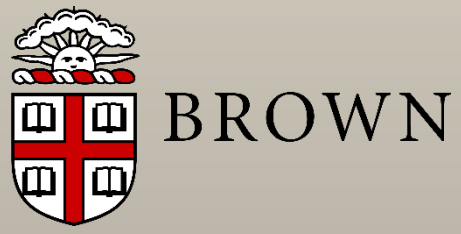
Marcelo Martins
Brown University



Justin Cappos
New York University

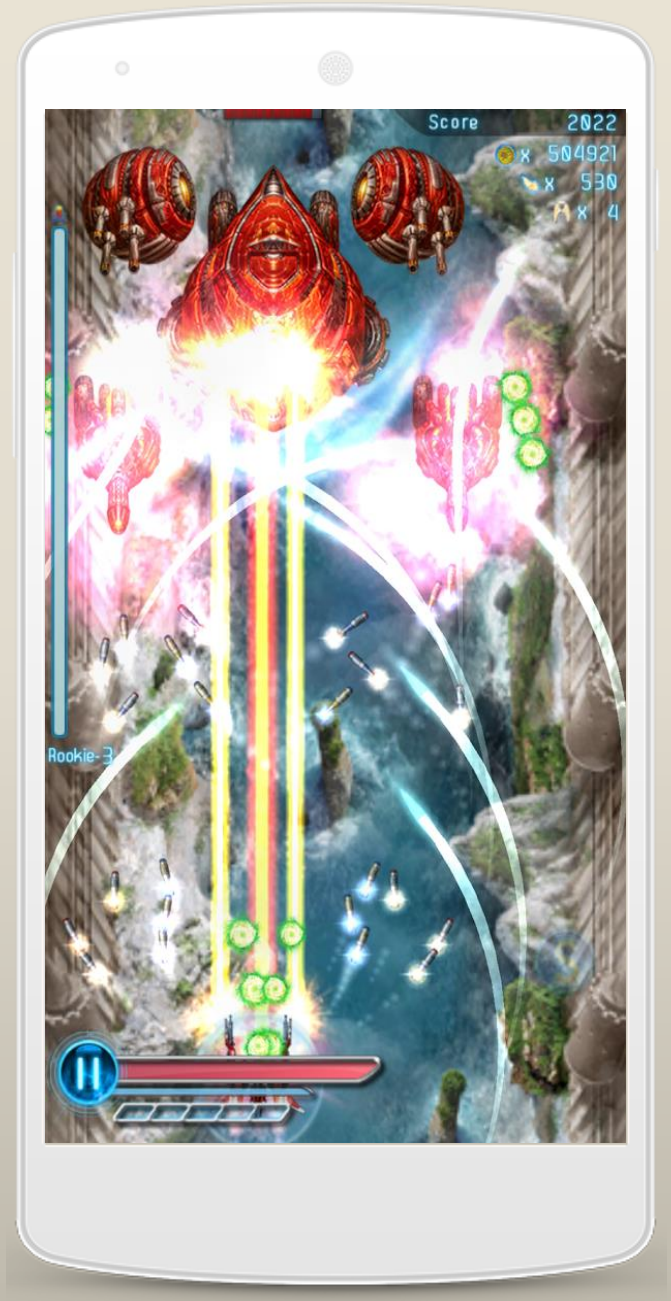


Rodrigo Fonseca
Brown University





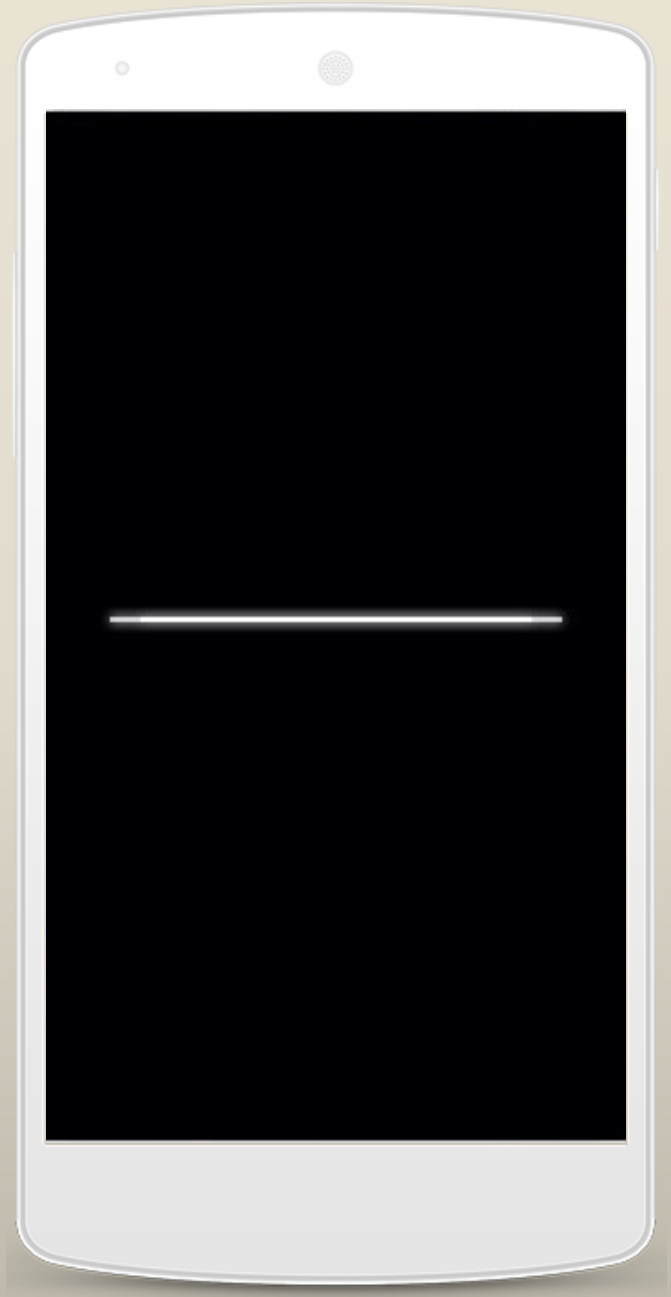

11:08
Google
01:38
Moon+ Reade
Connect charger
The battery is getting low.
1% remaining
Battery use
OK



Your Battery and You

Active device → high battery drainage

Matches our battery life expectation



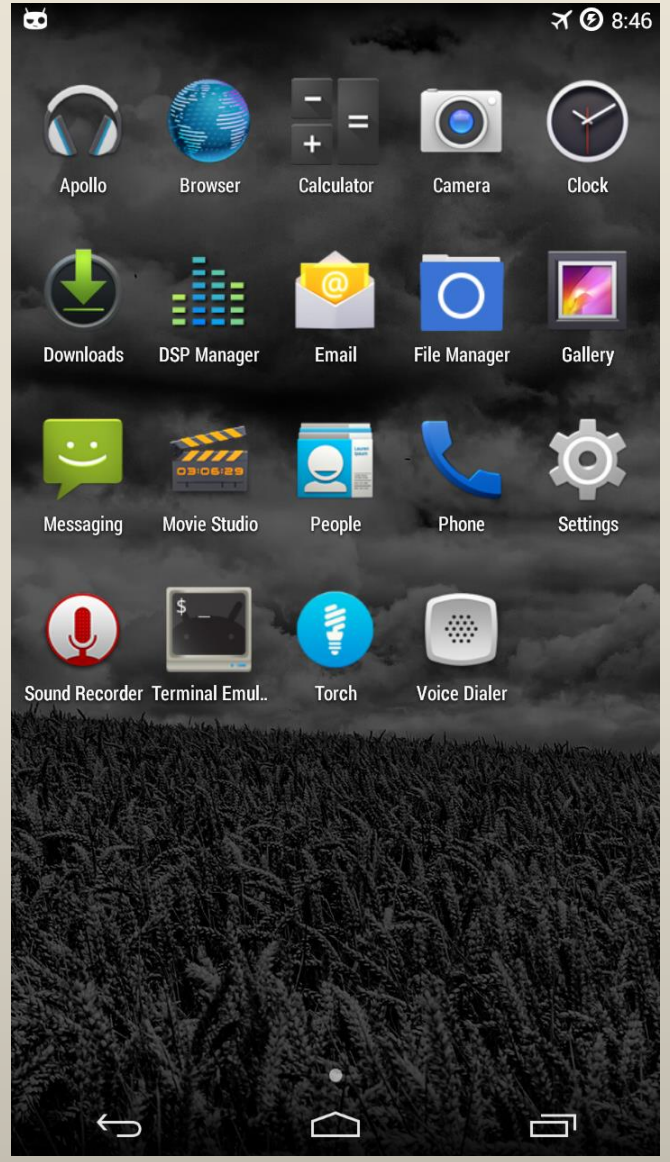
Your Battery and You

Idle device → expect low battery drain (?)

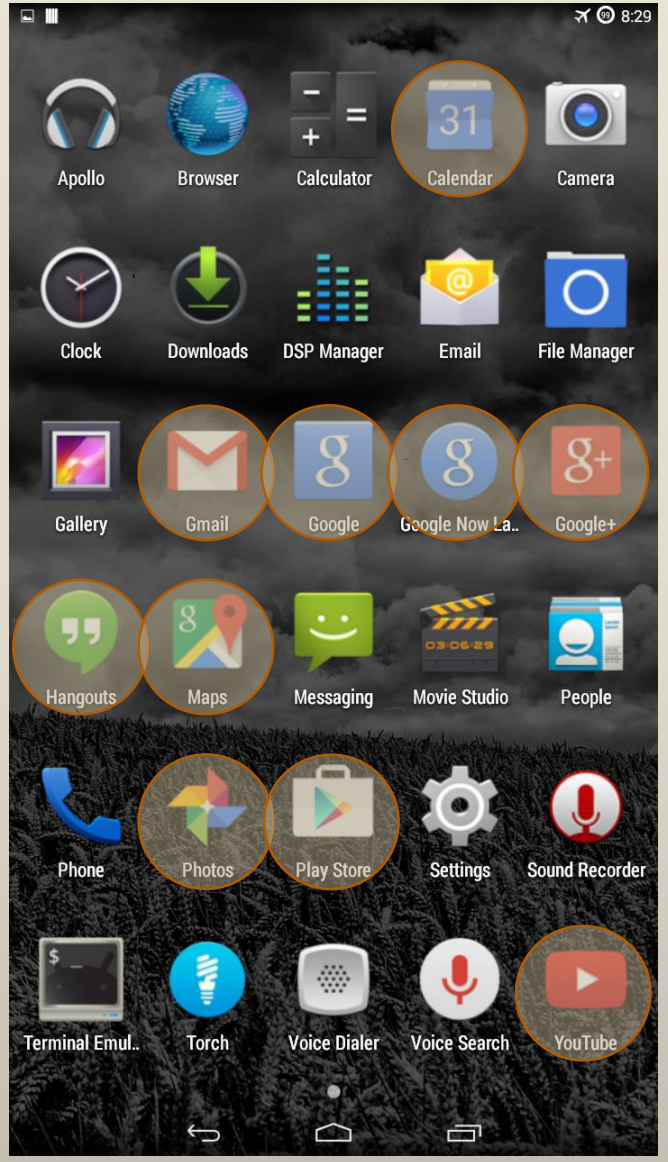
Background activities break low-power harmony

Energy bugs keep the device awake

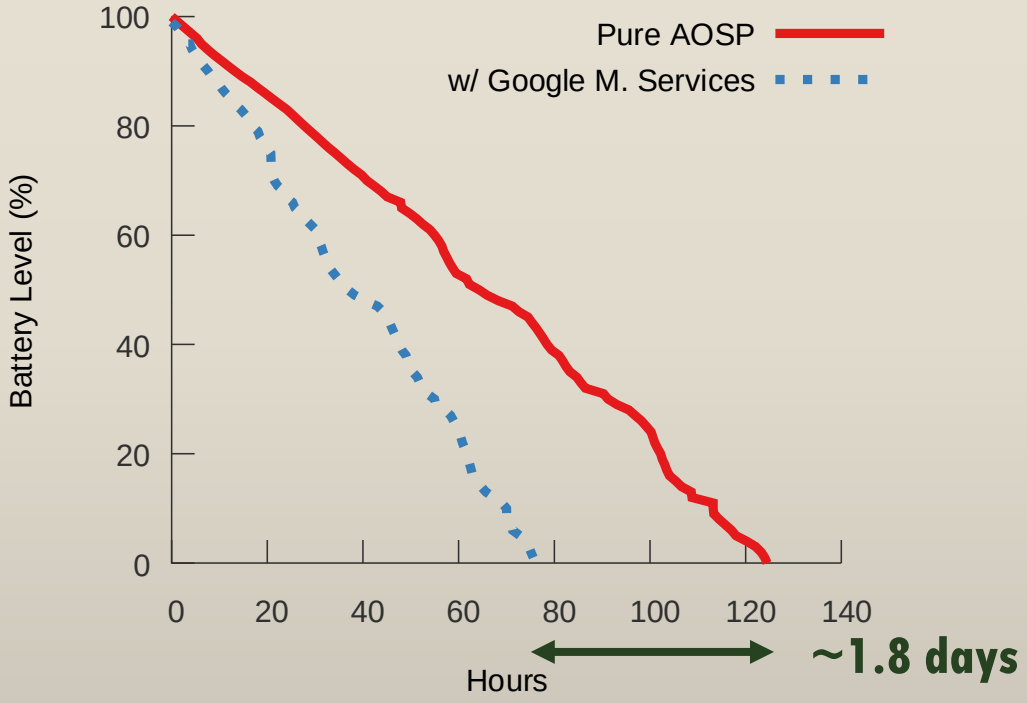
Hidden Energy Drain



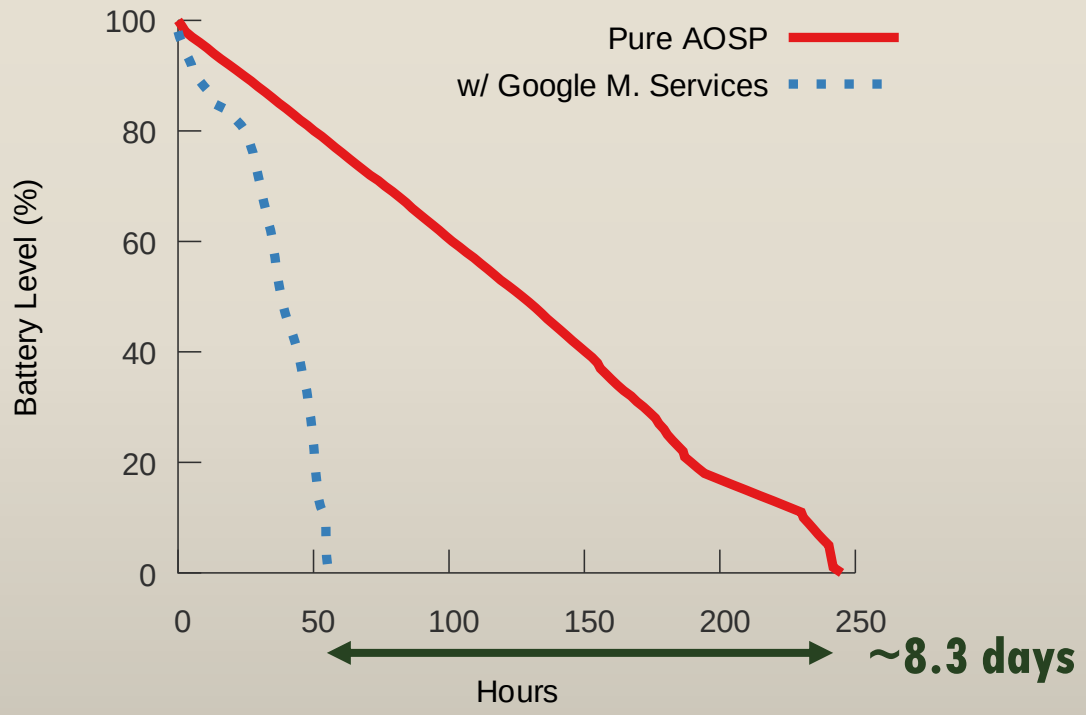
Pure AOSP



+ Google Mobile Services

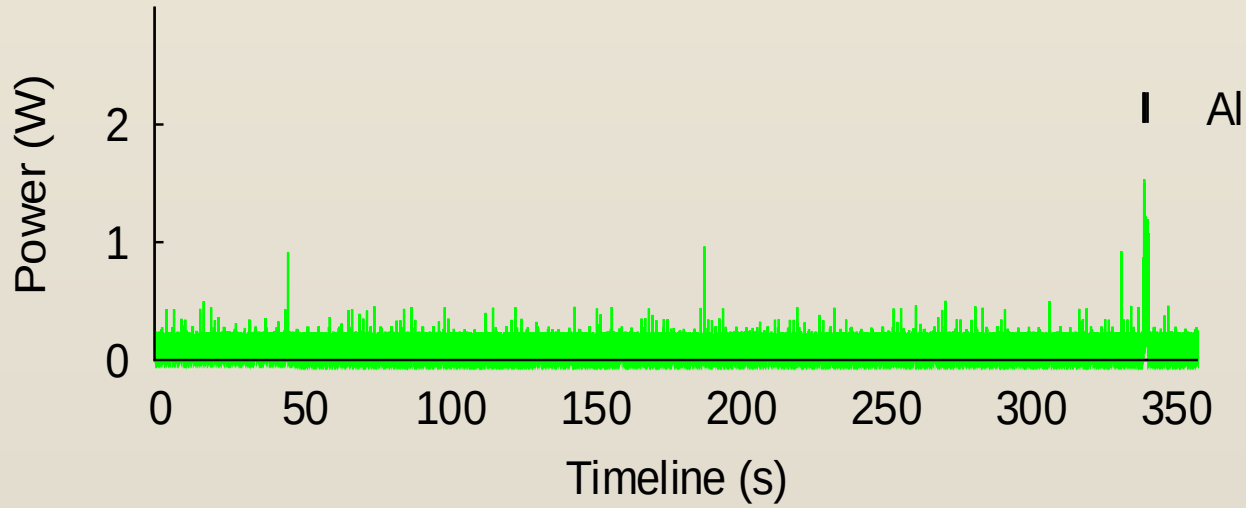


Samsung Galaxy Nexus

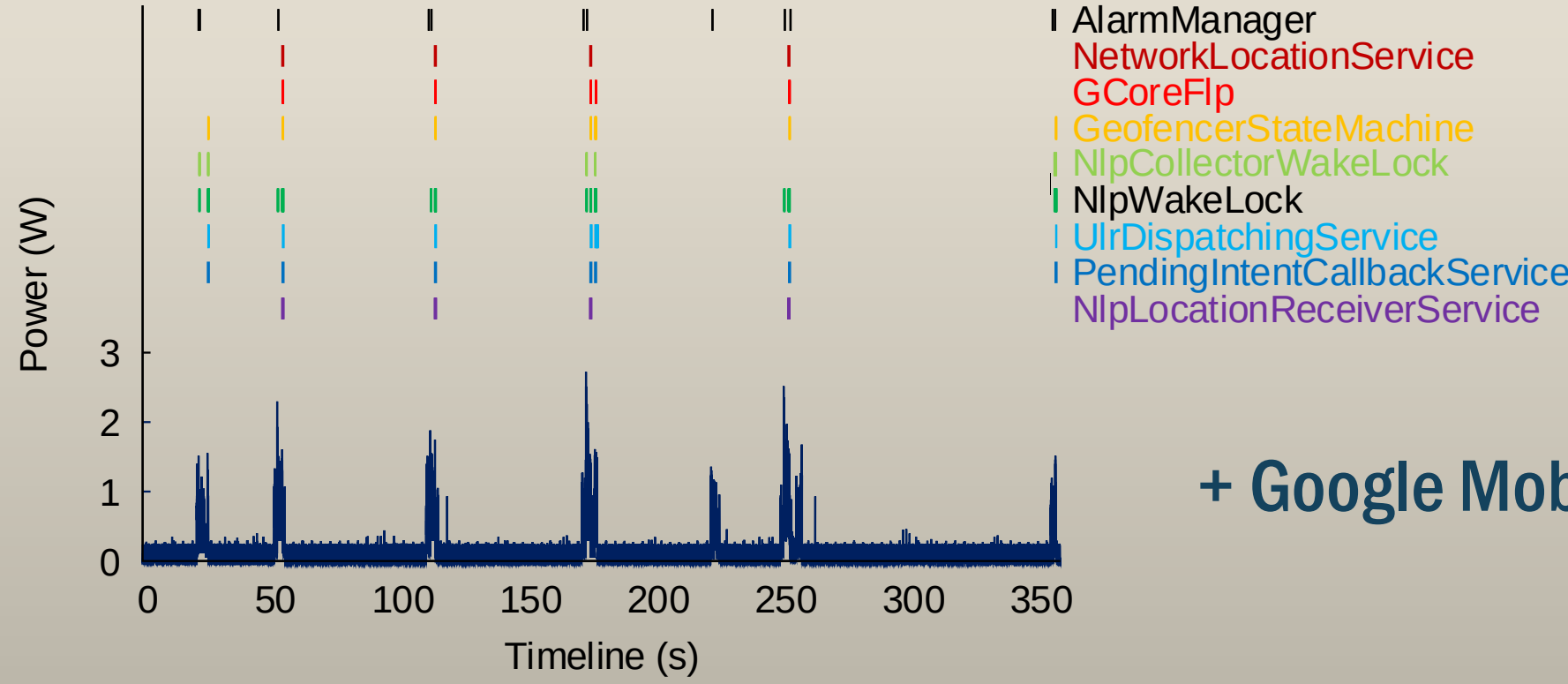


ASUS MeMO Pad 7

Hidden Energy Drain



Pure AOSP



+ Google Mobile Services

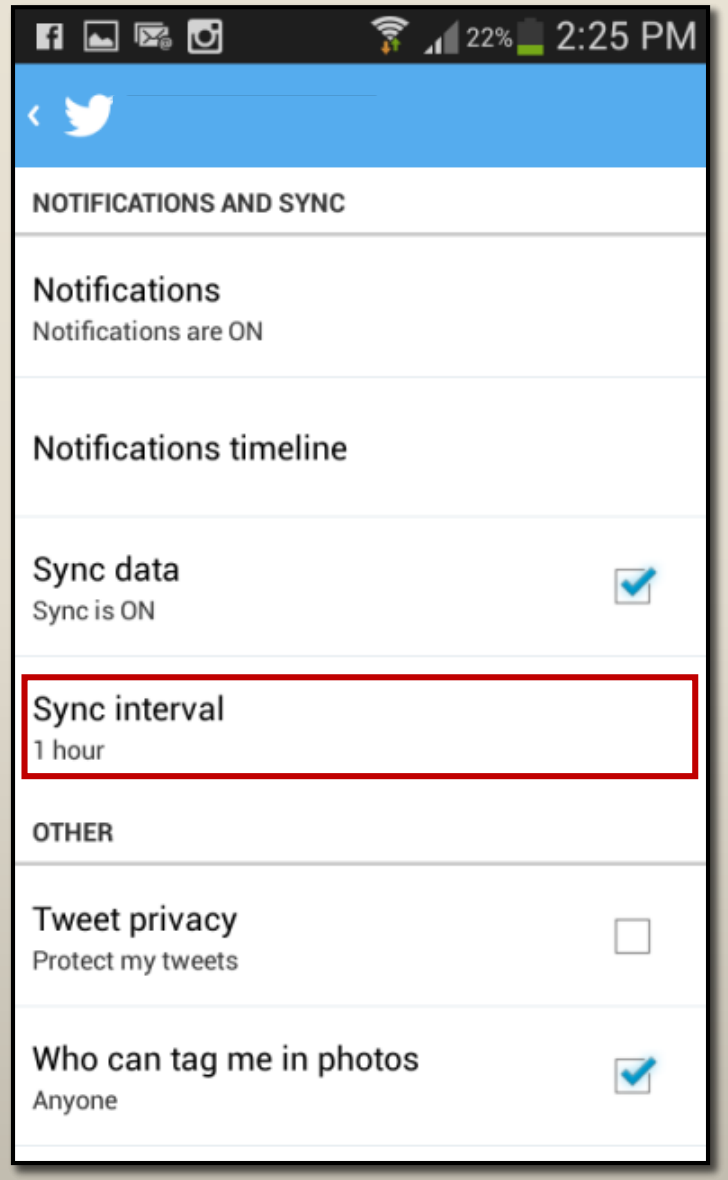
What Can We Do Today?

Reconfigure apps

Uninstall/Switch apps

Kill apps

Change system settings



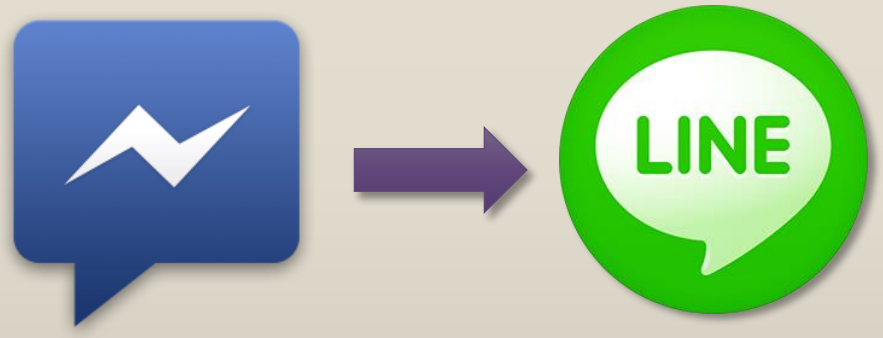
What Can We Do Today?

Reconfigure apps

Uninstall/Switch apps

Kill apps

Change system settings



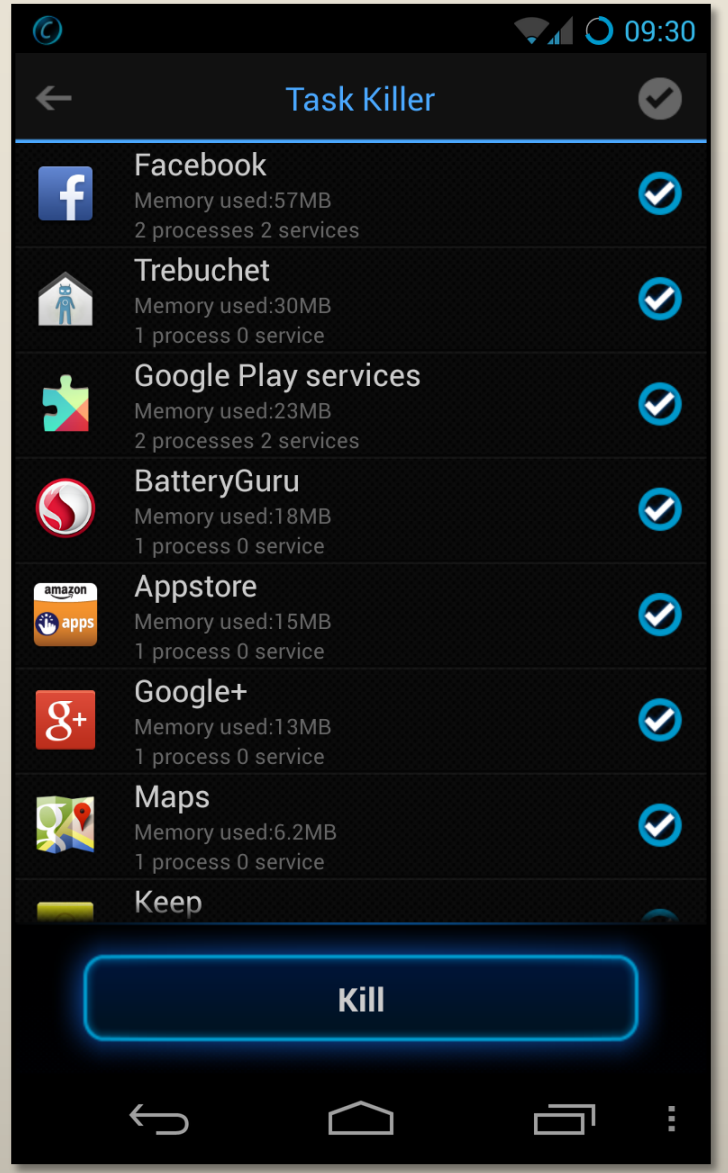
What Can We Do Today?

Reconfigure apps

Uninstall/Switch apps

Kill apps

Change system settings



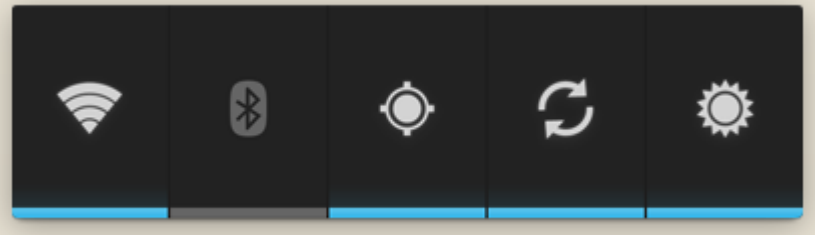
What Can We Do Today?

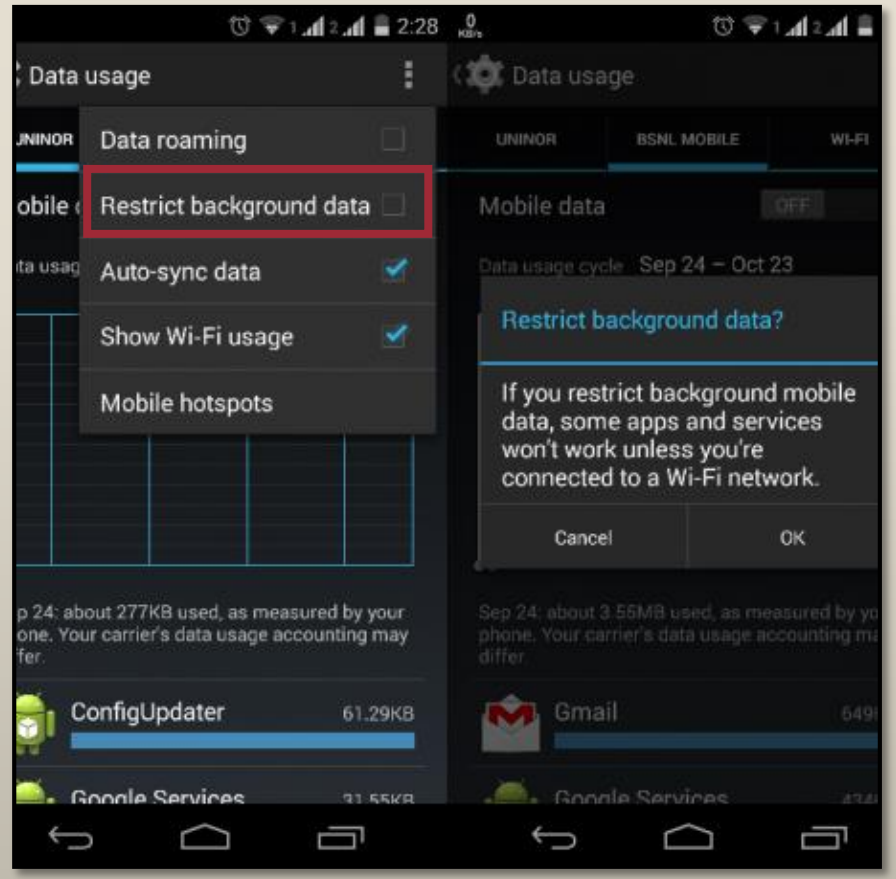
Reconfigure apps

Uninstall/Switch apps

Kill apps

Change system settings





TAMER



Runtime mechanism to rate-limit background events

Backgrounding Sources?

Services

Alarms

Intent Receivers

Wakelocks



Backgrounding Sources?

Services

Alarms

Intent Receivers

Wakelocks



Backgrounding Sources?

Services

Alarms

Intent Receivers

Wakelocks



Backgrounding Sources?

Services

Alarms

Intent Receivers

Wakelocks



Backgrounding Sources?

Services

Alarms

Intent Receivers

→ Asynchronous

Wakelocks

→ Symmetric



Prelude to Blocking: Event Dispatching




```
class fetchTemperature extends IntentService {
    @Override
    public void onHandleIntent(…) {…}
};
```

1. Event definition (what)

Prelude to Blocking: Event Dispatching




```
class fetchTemperature extends IntentService {…}

startService(new Intent(…, fetchTemperature.class));
```

1. Event definition (what)
2. Event request

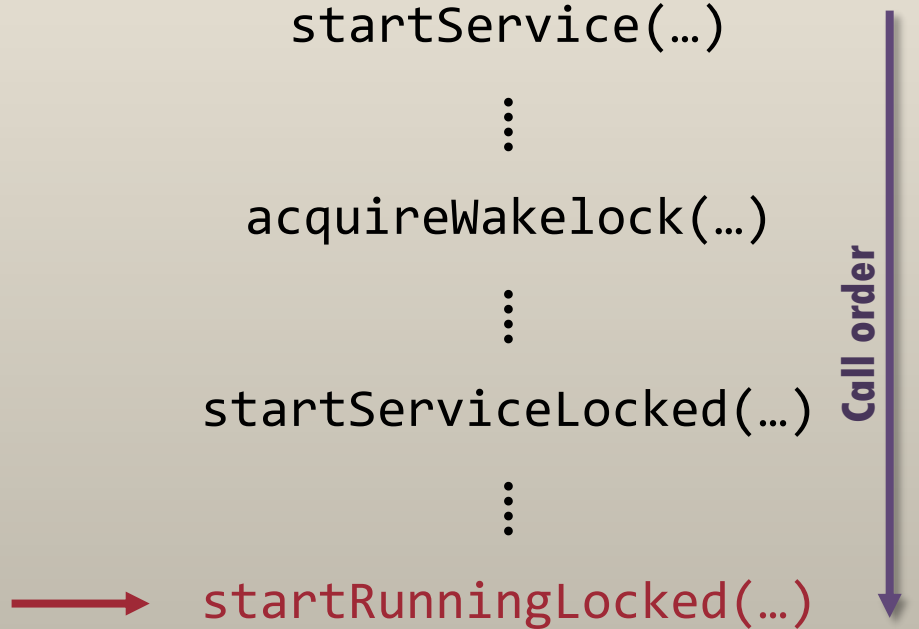
Prelude to Blocking: Event Dispatching




```
class fetchTemperature extends IntentService {…}

startService(new Intent(…, fetchTemperature.class));
```

1. Event definition (what)
2. Event request
3. Event callback (where)



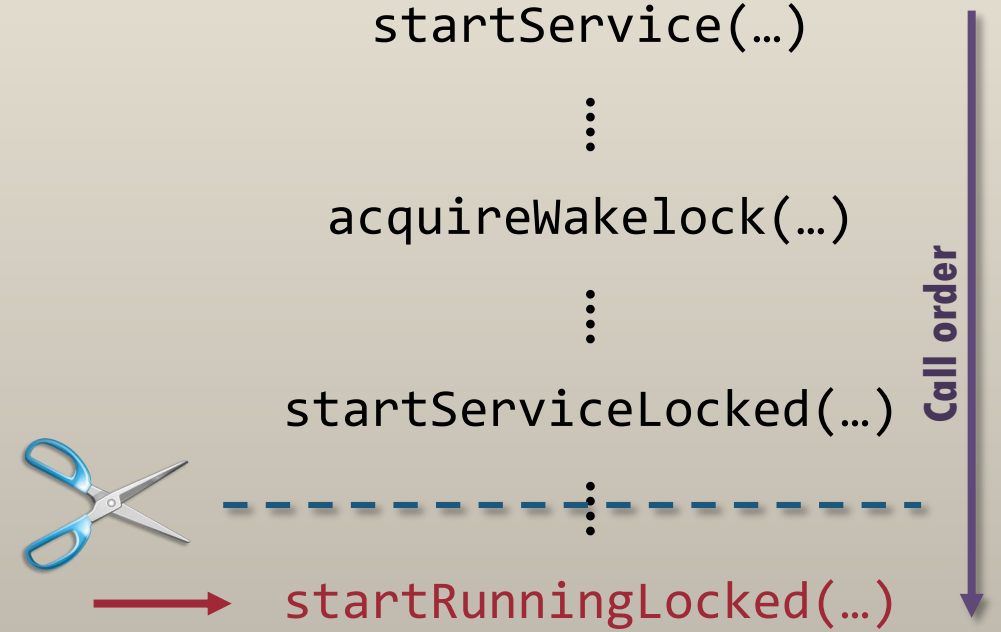
Blocking the Event Dispatch



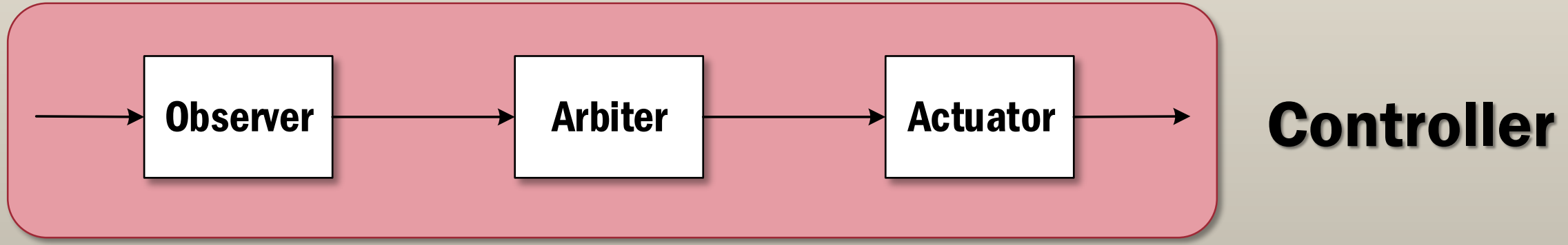

```
class fetchTemperature extends IntentService {…}

startService(new Intent(…, fetchTemperature.class));
```

1. Event definition (what)
2. Event request
3. Event callback (where)



Throttling Abstraction



App

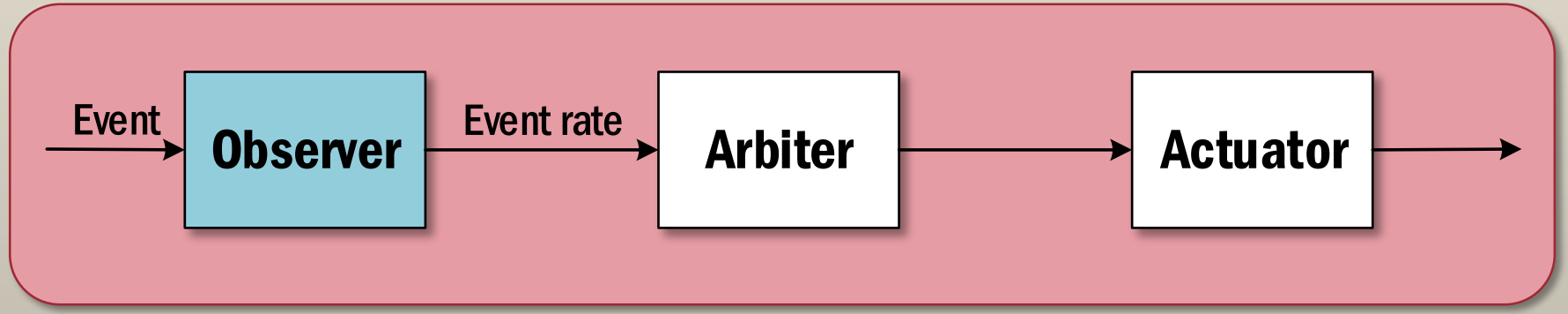
Tomorrow's temperature?



Tamer



App	Name	Type	#Calls
	fetchTemp	Service	1231+1
	syncPhotos	Service	342
	scanSystem	Service	3



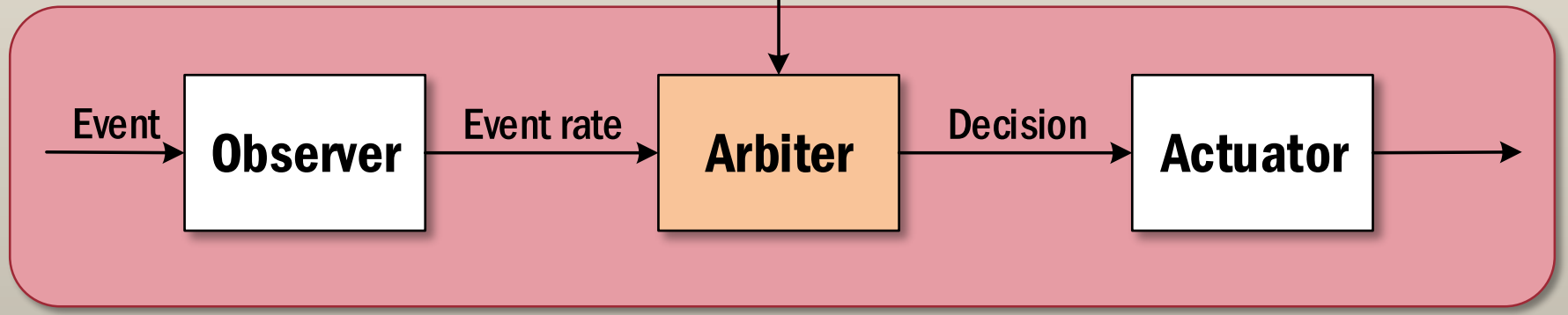
Controller



User

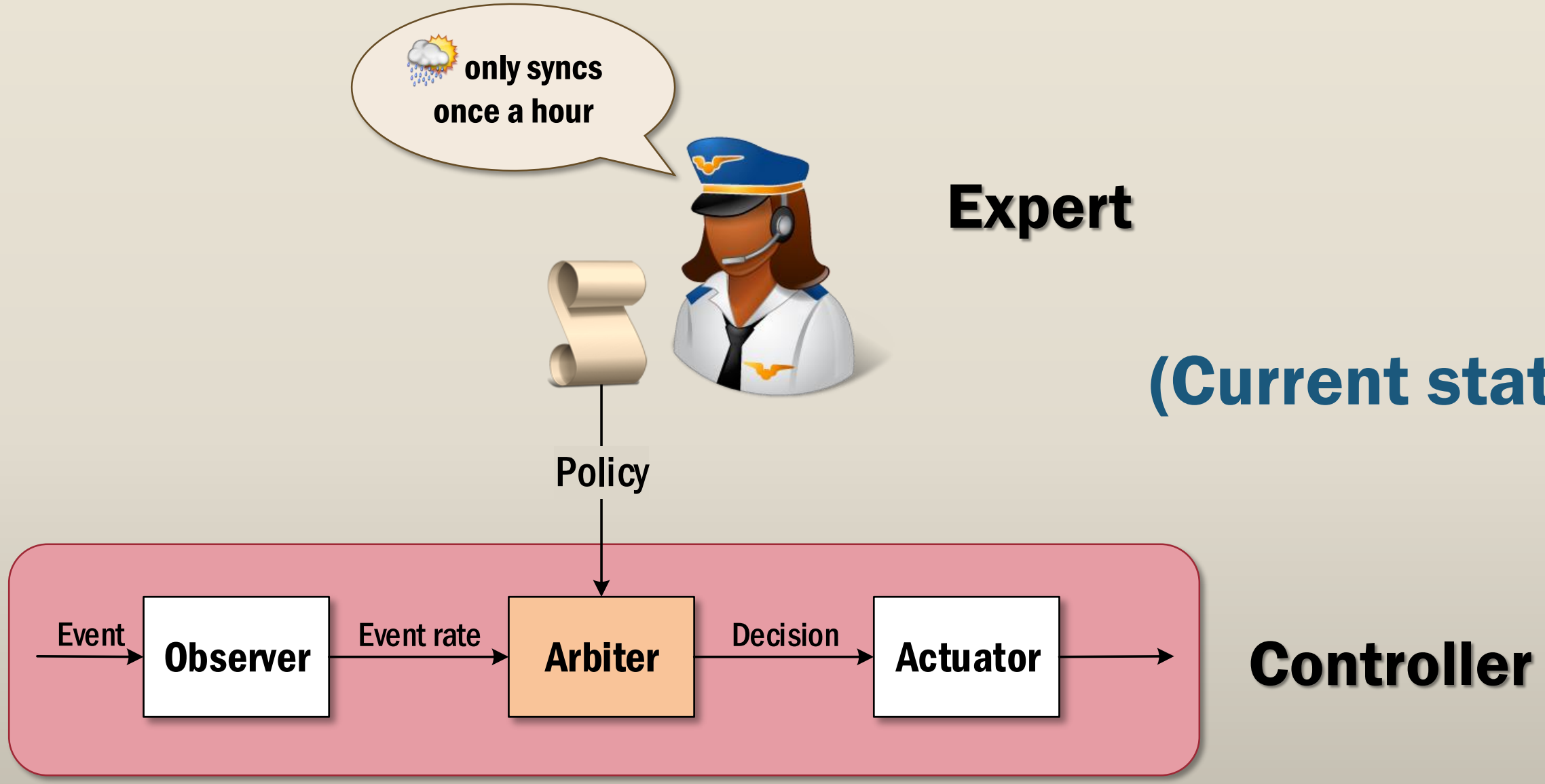
(Ideal case)

Policy



Controller

```
if (monitoredRate > policyThreshold) → should block
```




```
if (monitoredRate > policyThreshold) → should block
```

App

TAMER

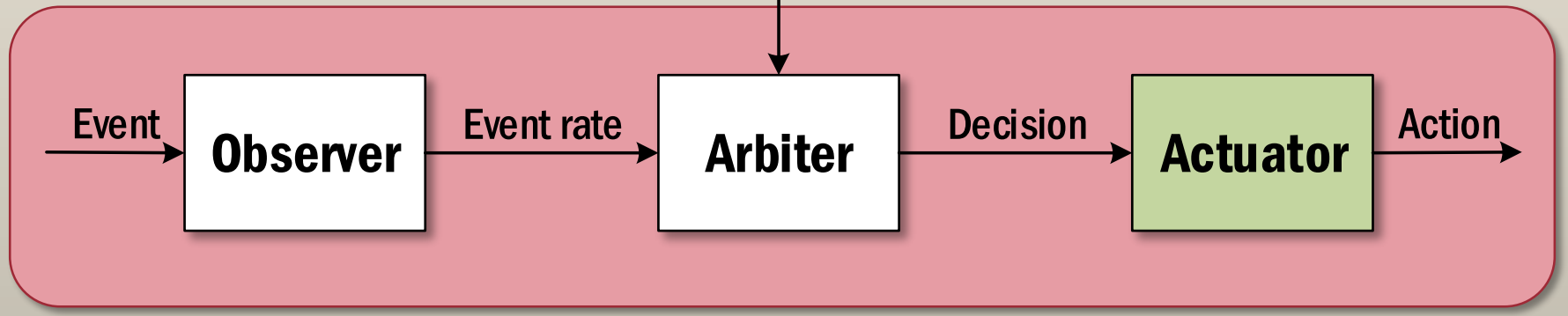
Tomorrow's temperature?



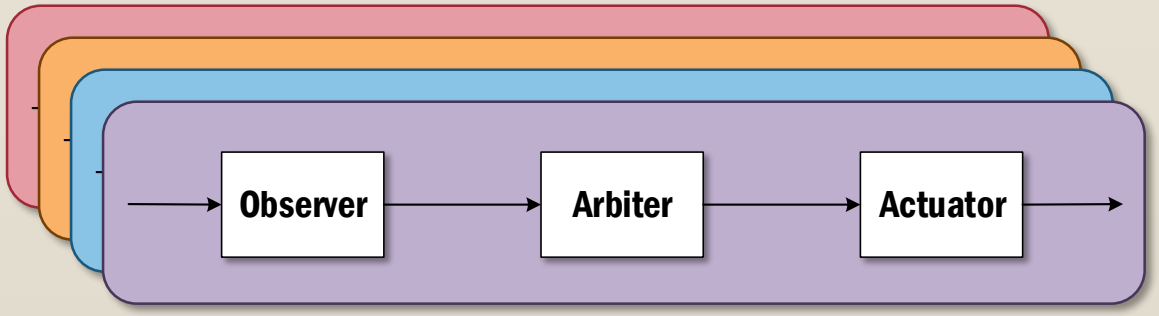




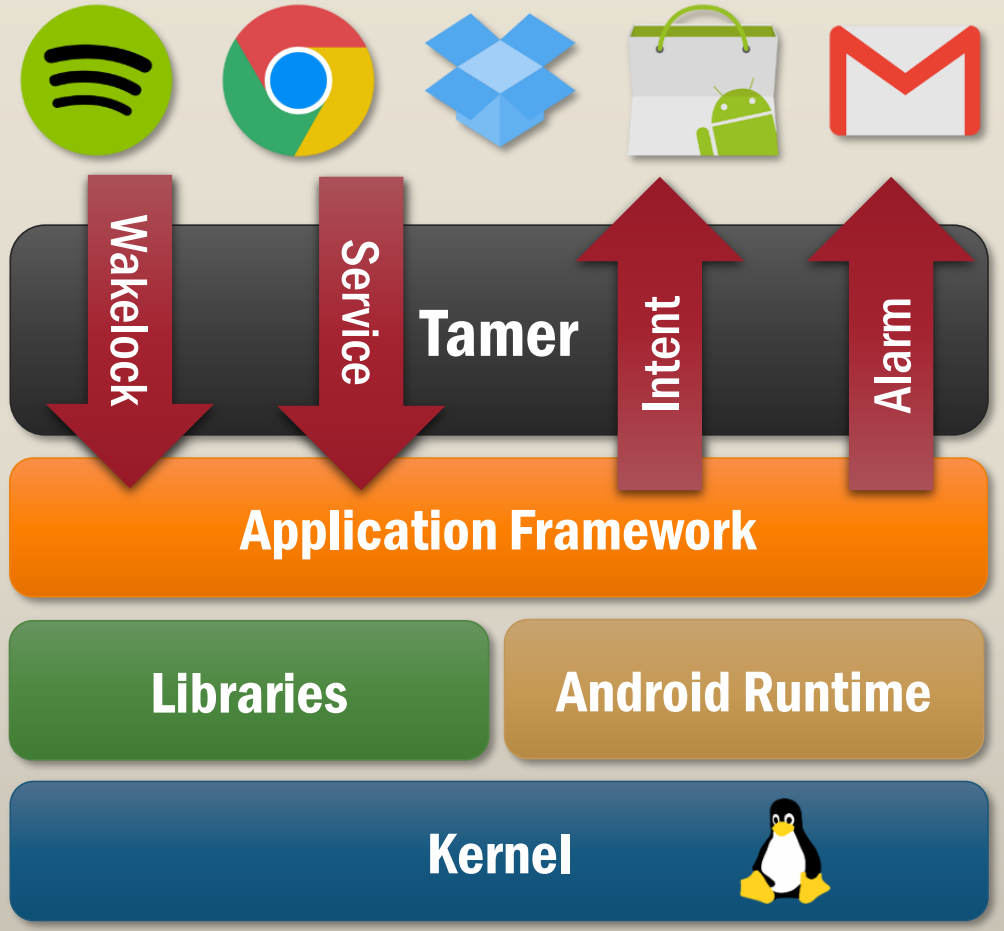
Policy



Controller



- Services
- Intent Receivers
- Alarms
- Wakelocks



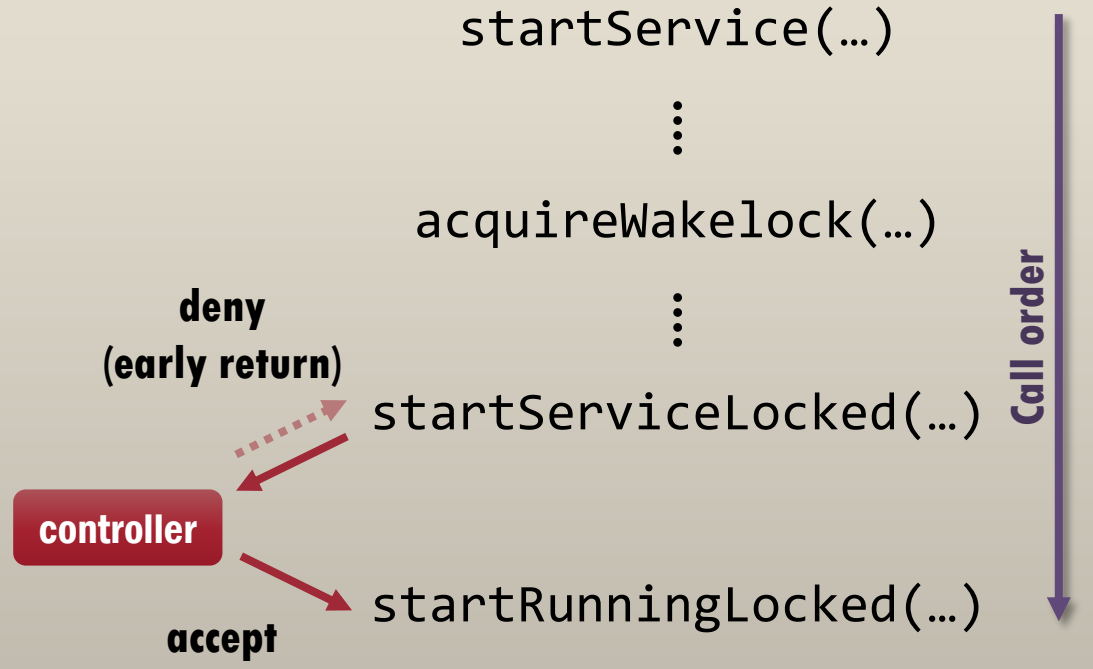
Implementation

Java-based runtime code injection via Xposed

Interposes on event callbacks to control their execution

Policies in JSON format





Installation

Installs like any other app (APK file)

Runtime modifications

- No need for app recompilation/reinstallation

Android compatibility

- Different vendors (Samsung, HTC, ASUS, CyanogenMod, etc.)
- Different versions (Ice Cream to KitKat)
- Requires rooting

Evaluation

Four devices

- Android KitKat

Experiments

- General lifetime
- Energy Bugs
- GPS throttling



Galaxy Nexus
(2011)



Kindle Fire2
(2012)



Galaxy S4
(2013)



MeMOPad 7
(2014)

Evaluation

Four devices

- Android KitKat

Experiments

- General lifetime
- Energy Bugs
- GPS throttling



Galaxy Nexus (2011)



Kindle Fire2 (2012)

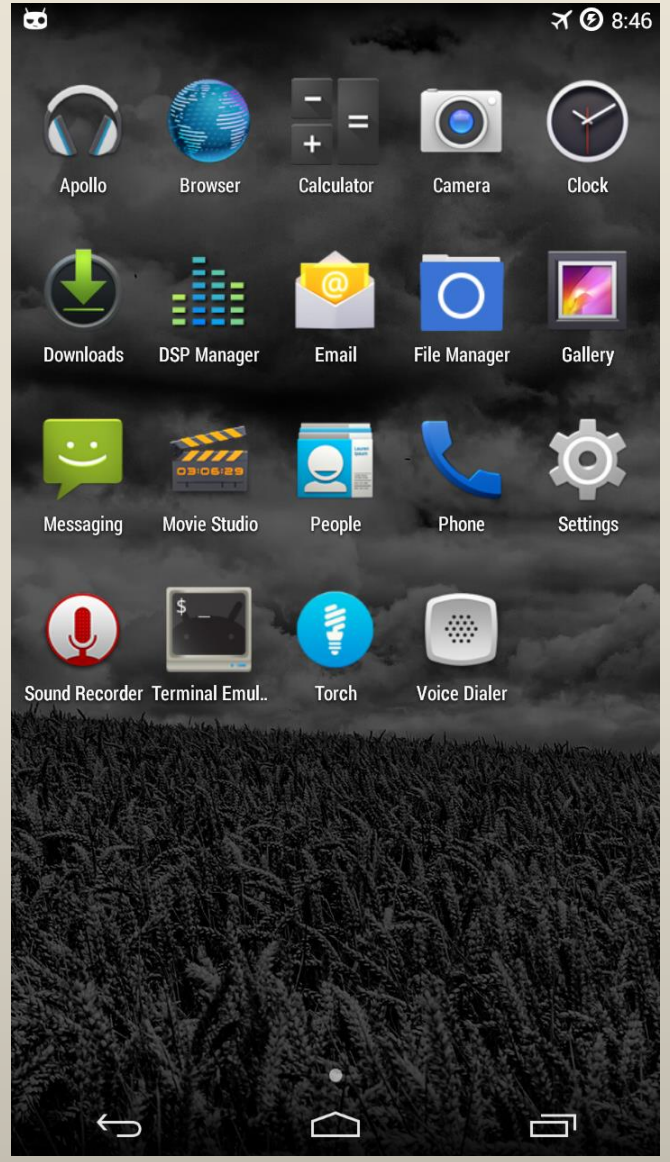


Galaxy S4 (2013)

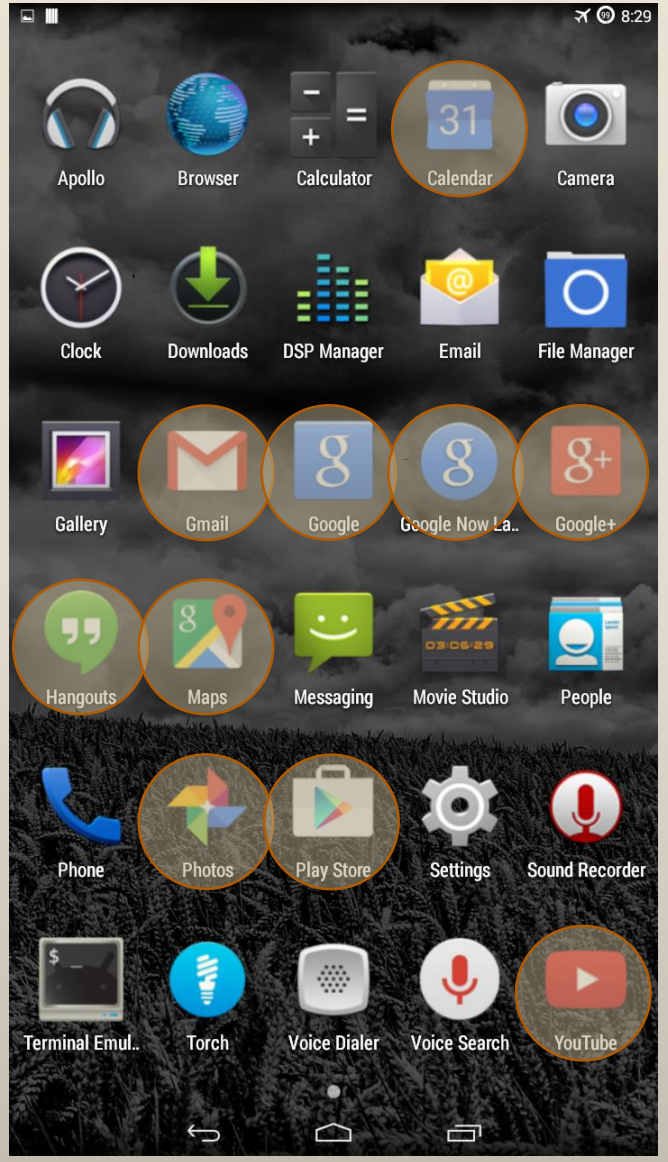


MeMOPad 7 (2014)

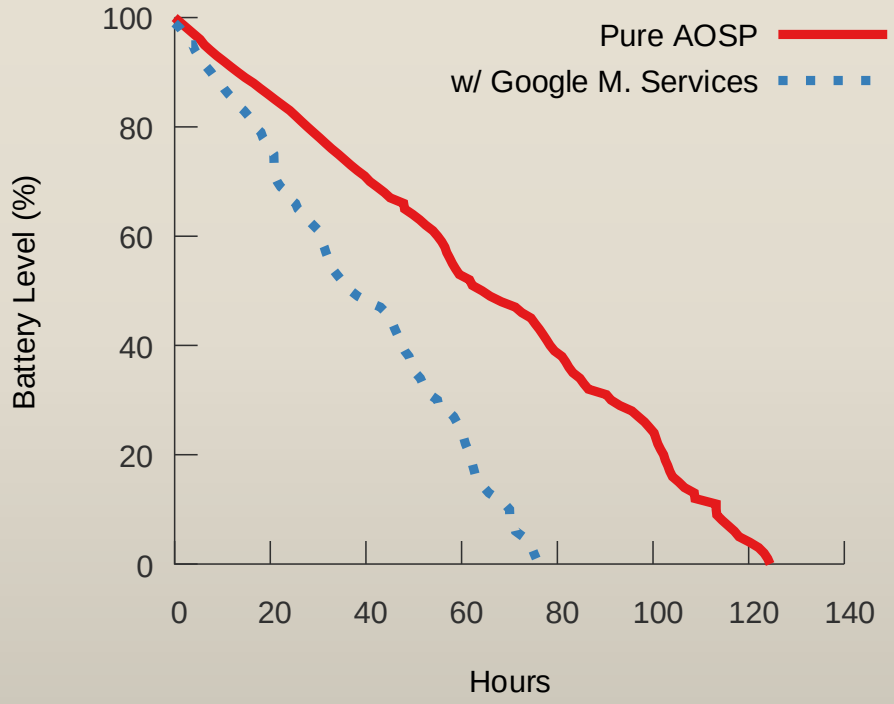
General Lifetime



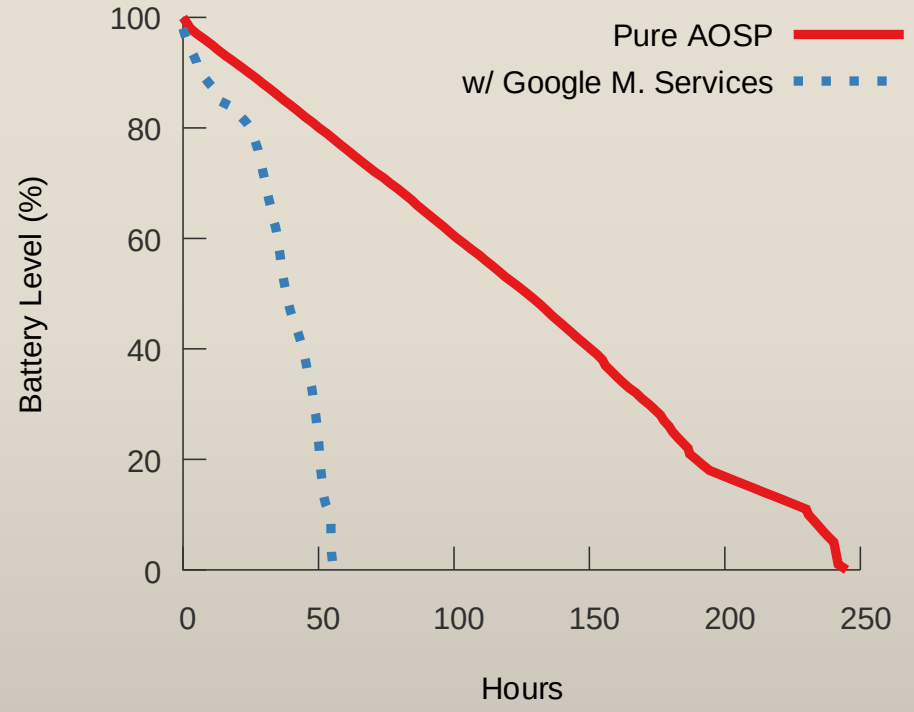
Pure AOSP



+ Google Mobile Services

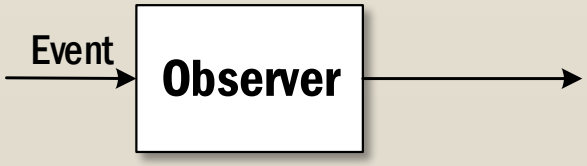


Samsung Galaxy Nexus



ASUS MeMO Pad 7

Exploring Taming Possibilities



Monitoring time: 100 hours

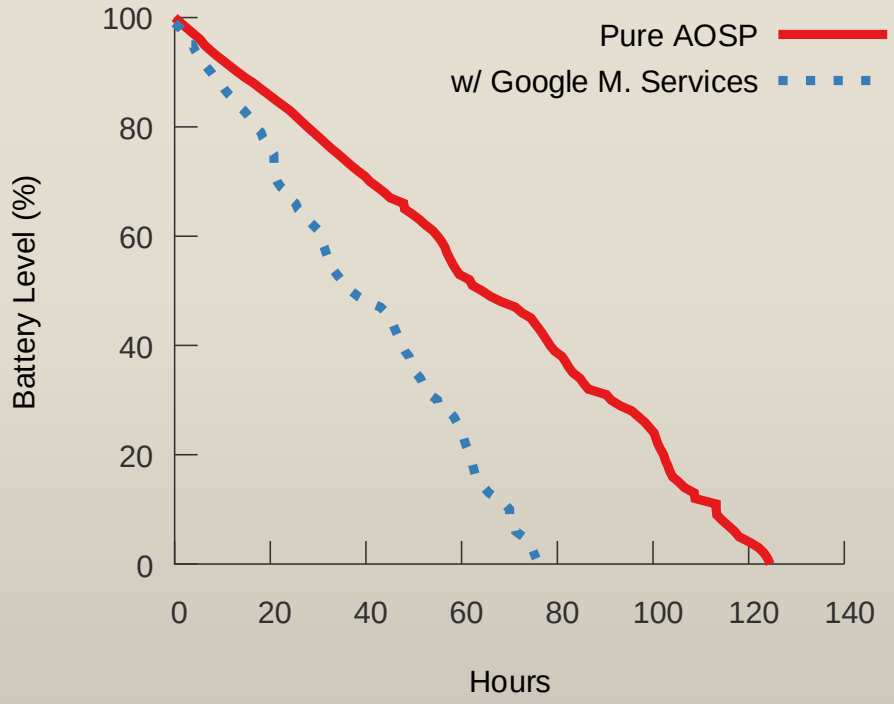
Event Name	Type	Count	Duration (s)
NlpWakelock	W	5963	1662.71
NlpCollectorWakelock	W	2121	3926.63
LocationManagerService	W	2030	67.12
NlpLocationReceiverService	S	1159	-
NetworkLocationService	S	579	-

One call per min
Sys awake 11hs

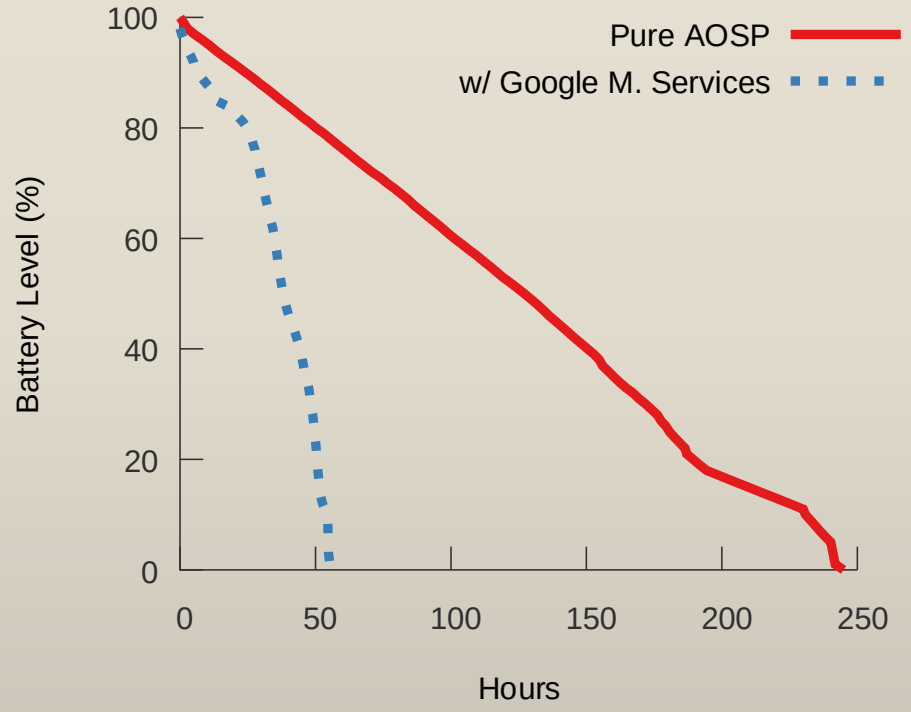
New Policies

Tamer-15: Events allowed every 15 minutes

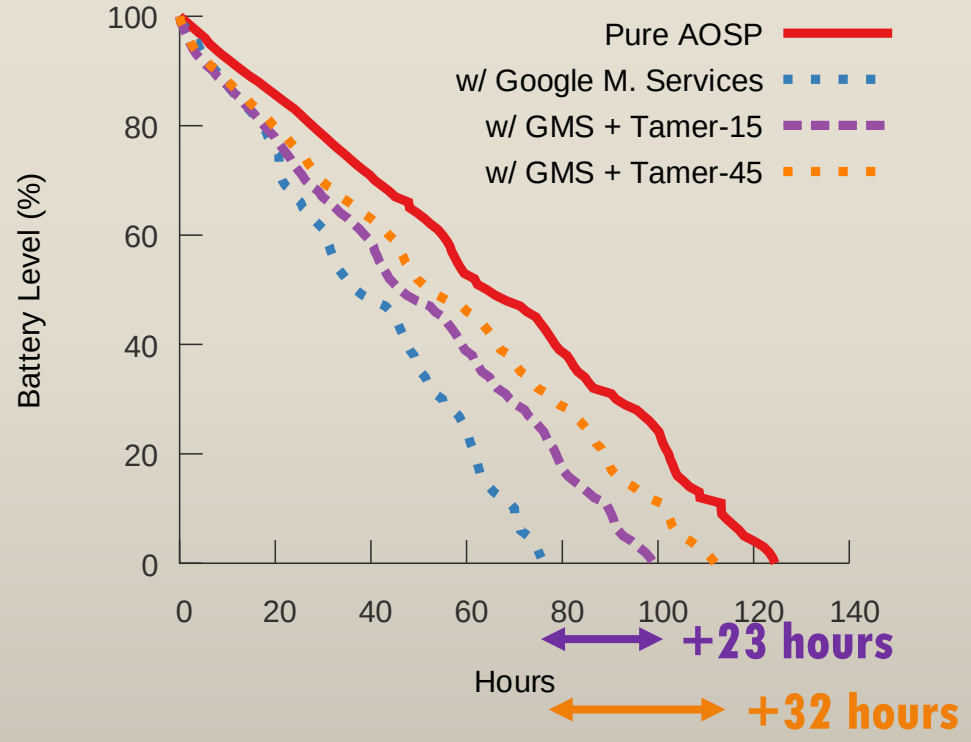
Tamer-45: Events allowed every 45 minutes



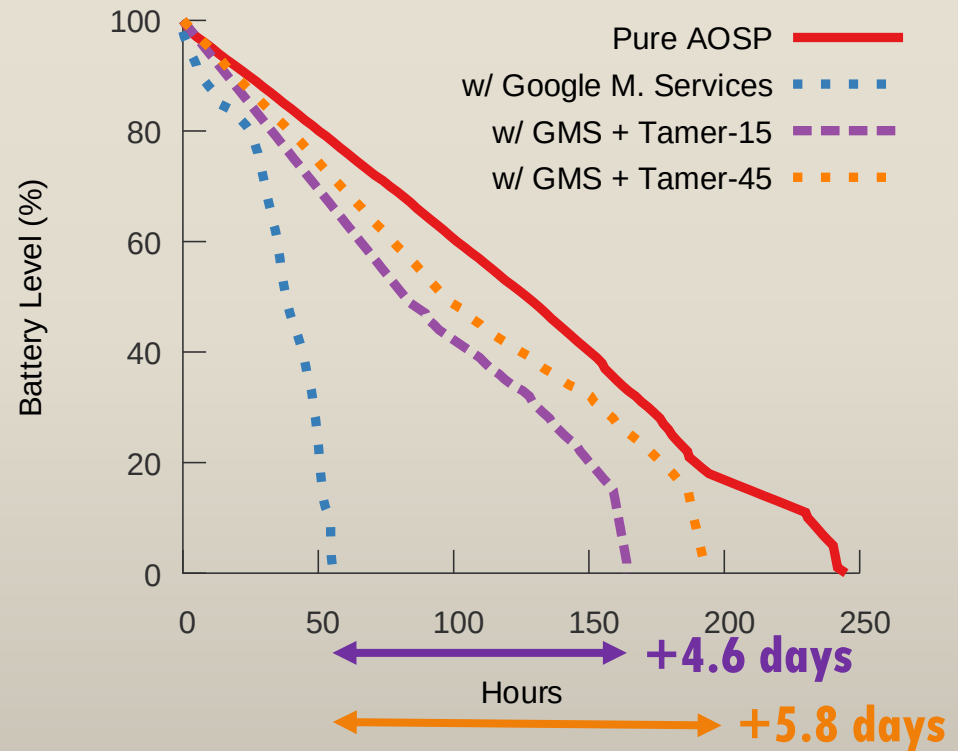
Samsung Galaxy Nexus



ASUS MeMO Pad 7

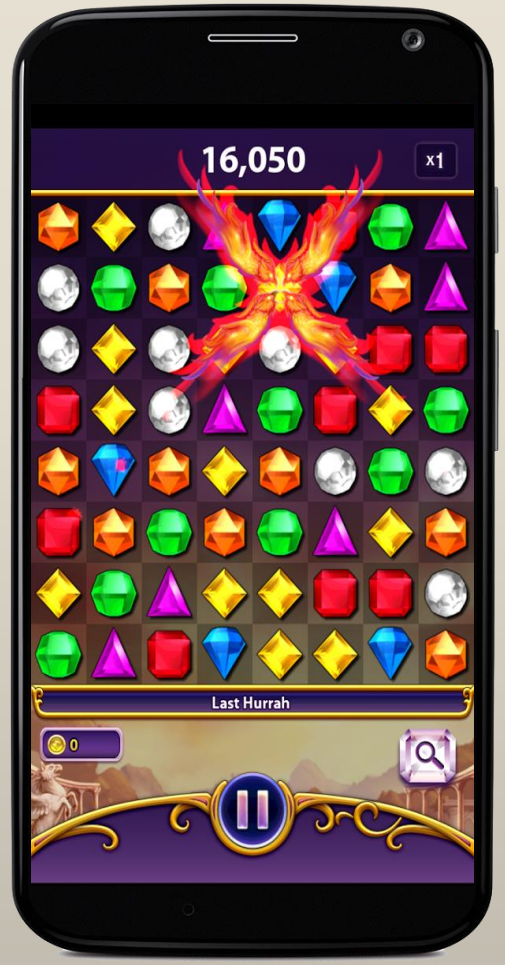


Samsung Galaxy Nexus



ASUS MeMO Pad 7

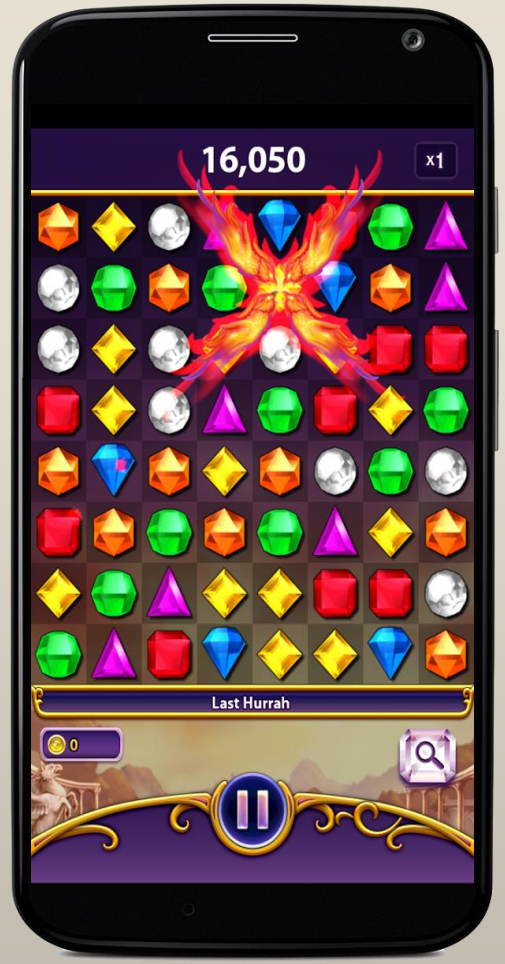
Chasing Energy Bugs – Bejeweled Blitz



10M+ downloads on Play Store

Top 20 on Puzzle Category

Chasing Energy Bugs – Bejeweled Blitz



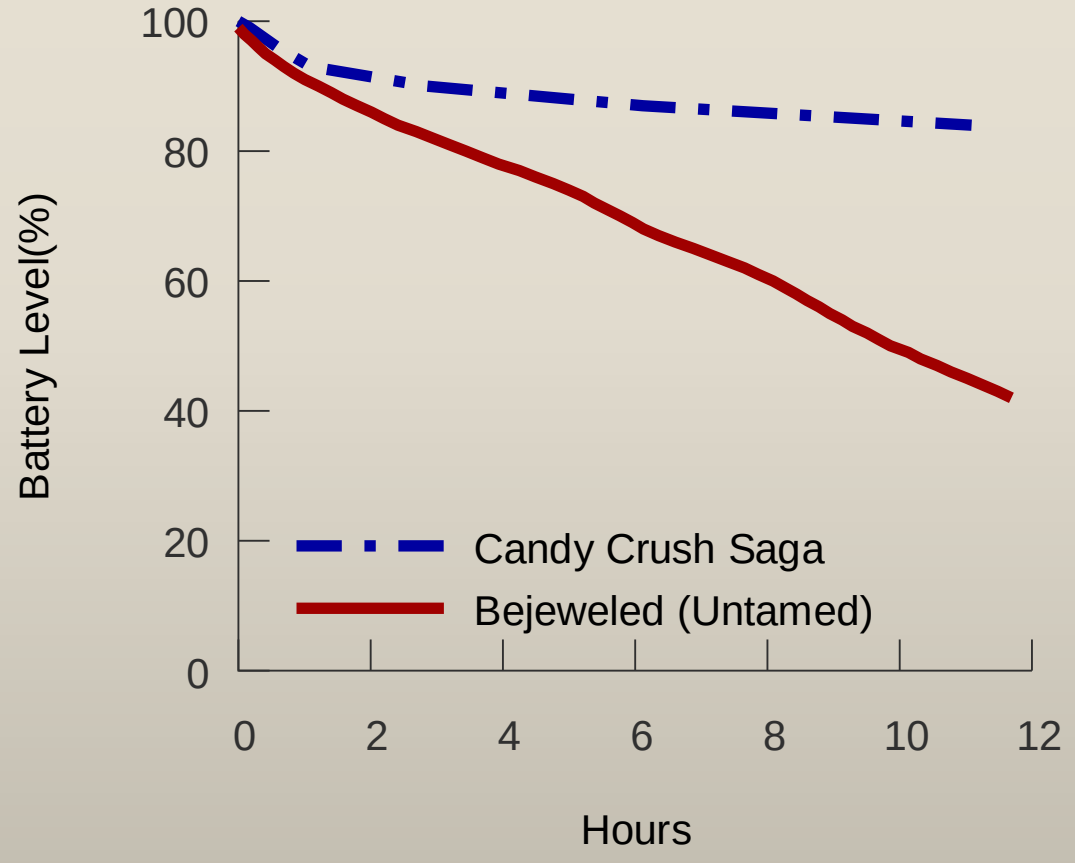


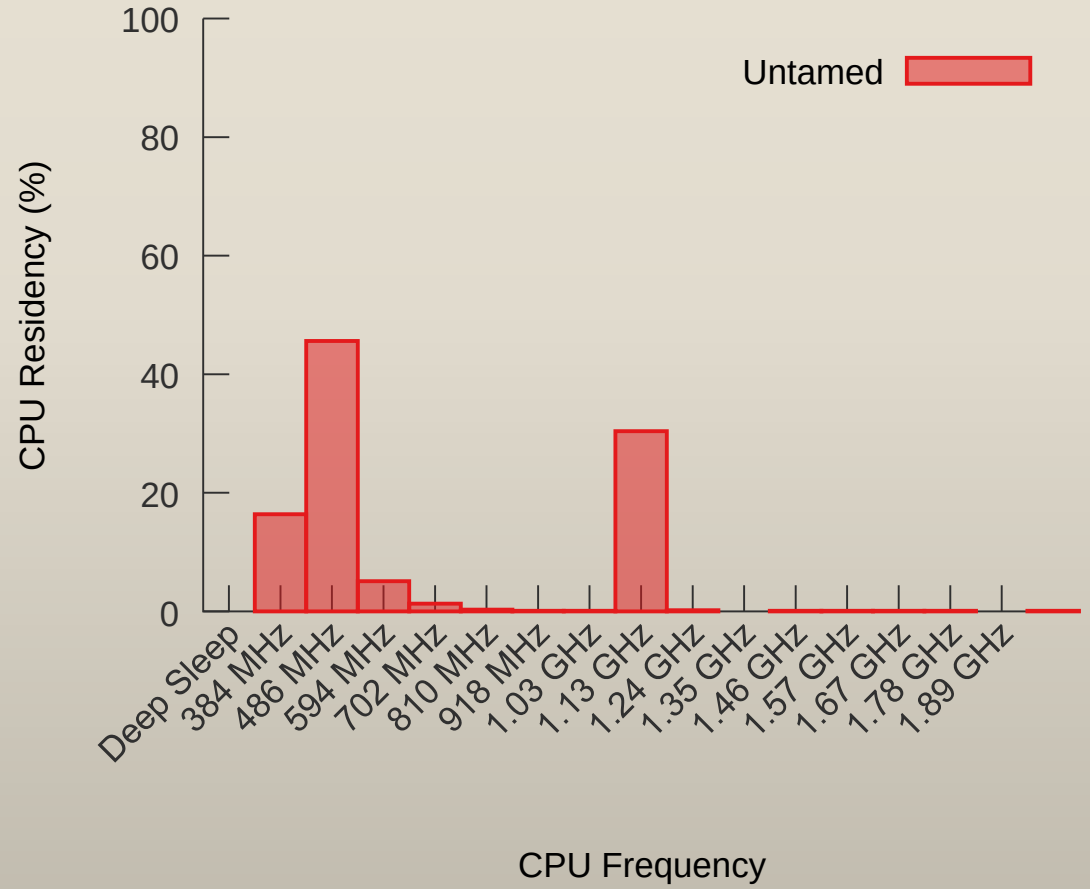


AudioIn

AudioIn (!!)

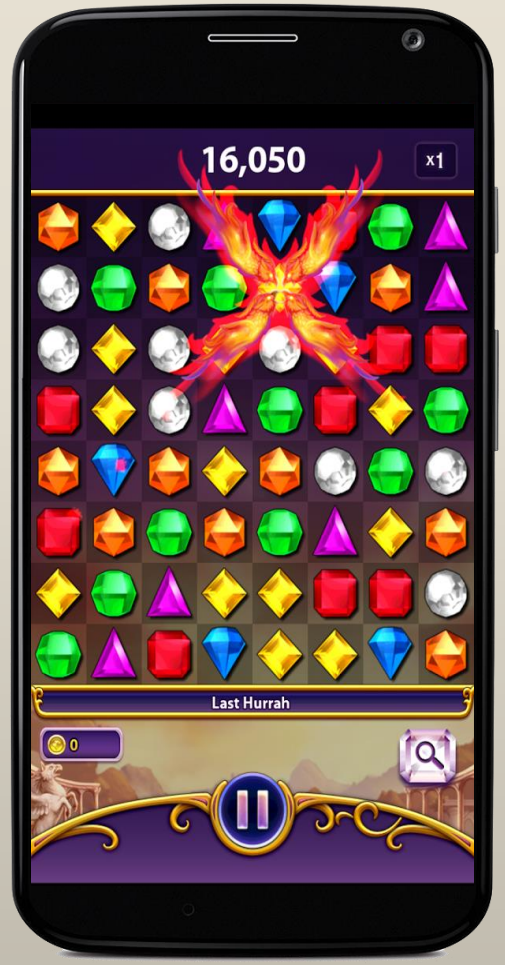
Bejeweled Blitz – Effects on Battery Lifetime





Samsung Galaxy S4

Chasing Energy Bugs – Bejeweled Blitz

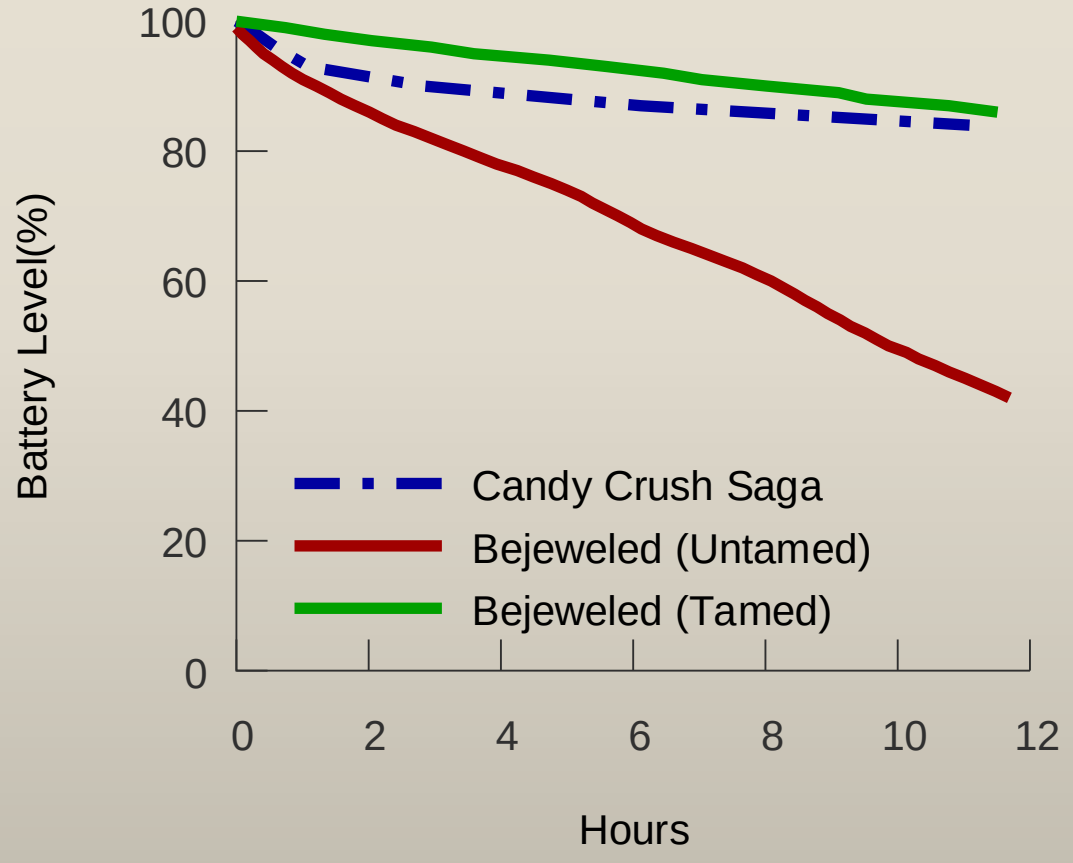


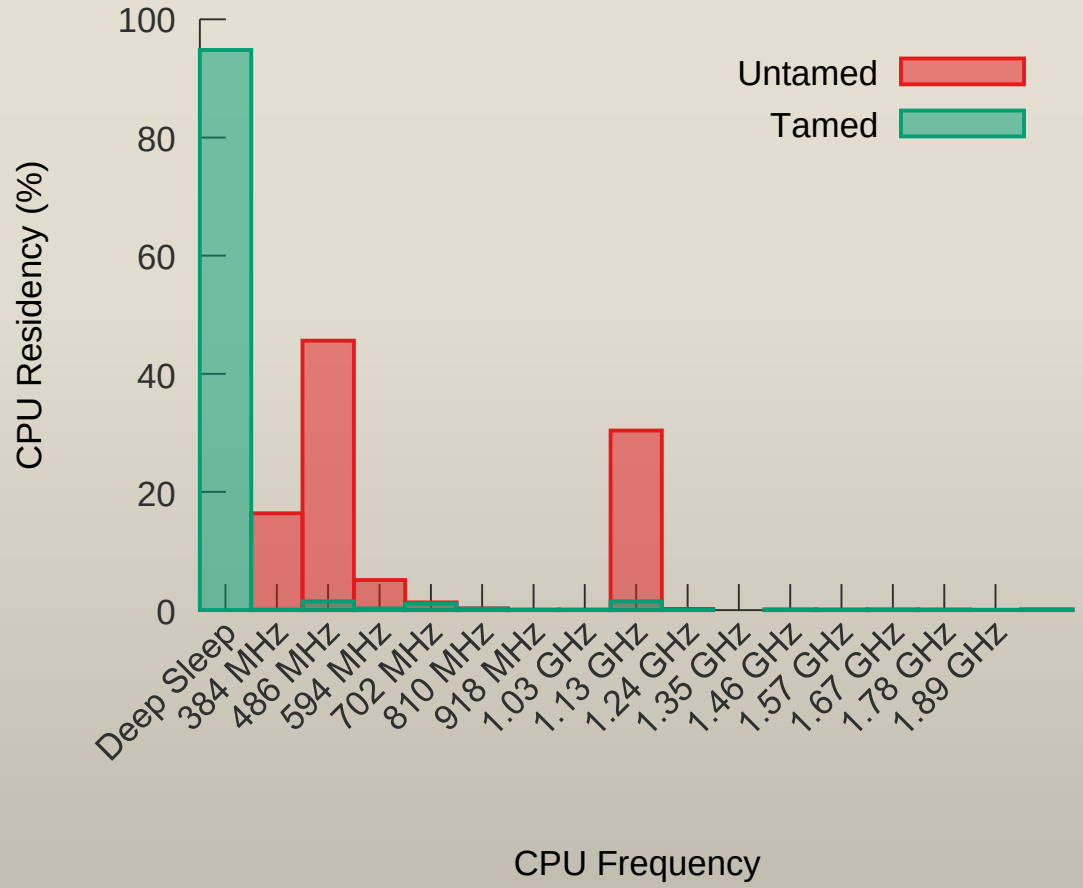


AudioIn

```
{
    “AppName”: “com.ea.BejeweledBlitz_na”
        “EventList”: {
            “Entry”: {
                “Type”: “wakelock”
                “Name”: “AudioIn”
                “AllowEvery”: “0”
                “BackgroundOnly”: “true”
            }
    }
}
```

After Taming





Samsung Galaxy S4

Limitations and Extensions

How to define good policies?

- Avoid breaking functionality
- Dynamic policies
- Context awareness



Limitations and Extensions

How to define good policies?

How to engage users?



Limitations and Extensions

How to define good policies?

How to engage users?

Event batching instead of blocking

Wrap Up

Mobile apps can silently kill battery

TAMER: OS-level mechanism to **throttle** background events

Separation of control and policy

Extends **battery** life / Selectively keeps **functionality**

Effectively mitigates energy bugs