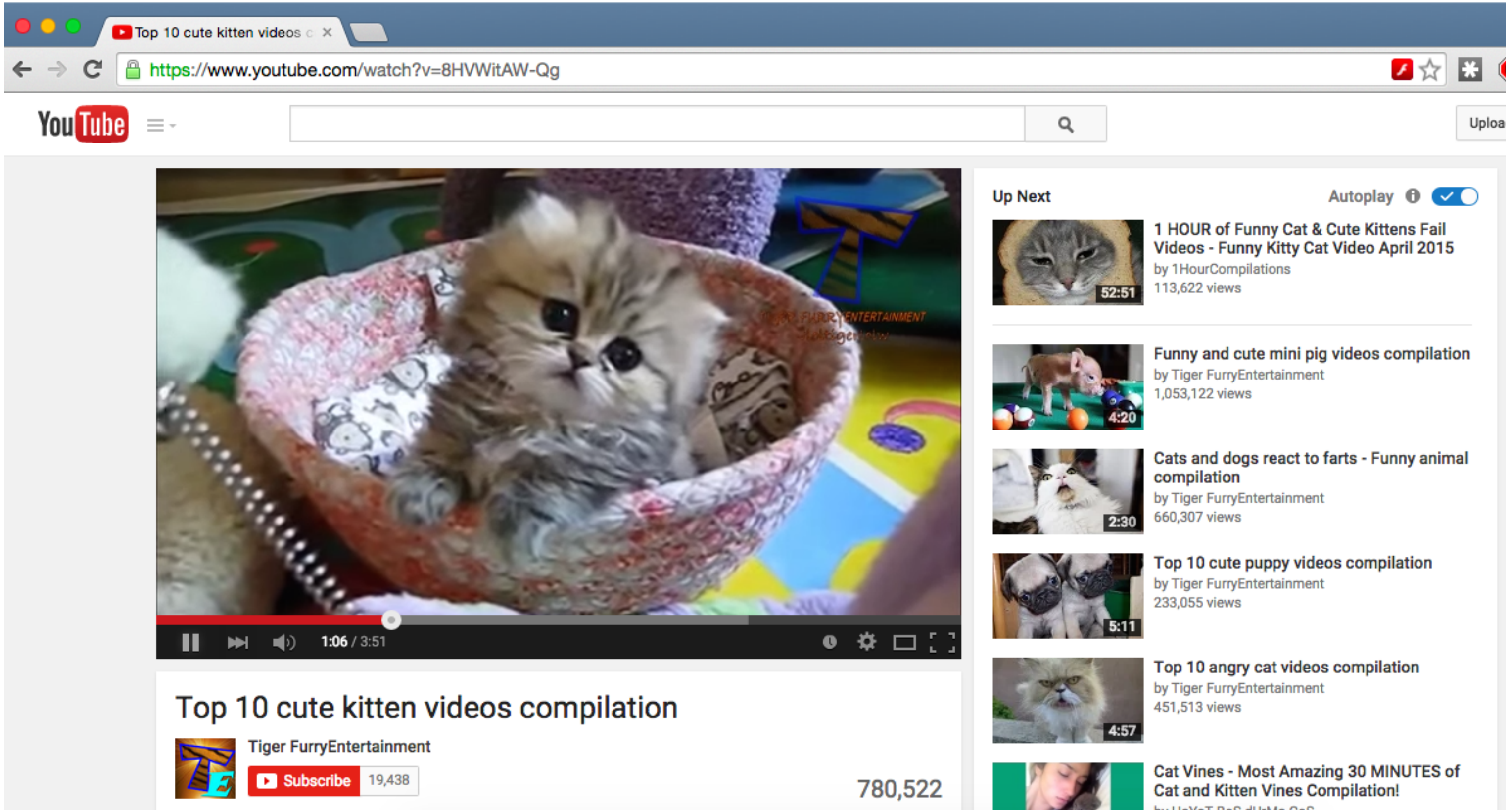
Fence: Protecting Device Availability With Uniform Resource Control

Tao Li†, Albert Rafetseder†, Rodrigo Fonseca*,
Justin Cappos†
†New York University *Brown University

Motivation



Top 10 cute kitten videos c
https://www.youtube.com/watch?v=8HVWitAW-Qg
YouTube
Uploa
1:06 / 3:51
Up Next
Autoplay
Subscribe
19,438

Top 10 cute kitten videos compilation

Tiger FurryEntertainment

780,522

1 HOUR of Funny Cat & Cute Kittens Fail Videos - Funny Kitty Cat Video April 2015
by 1HourCompilations
113,622 views
52:51

Funny and cute mini pig videos compilation
by Tiger FurryEntertainment
1,053,122 views
4:20

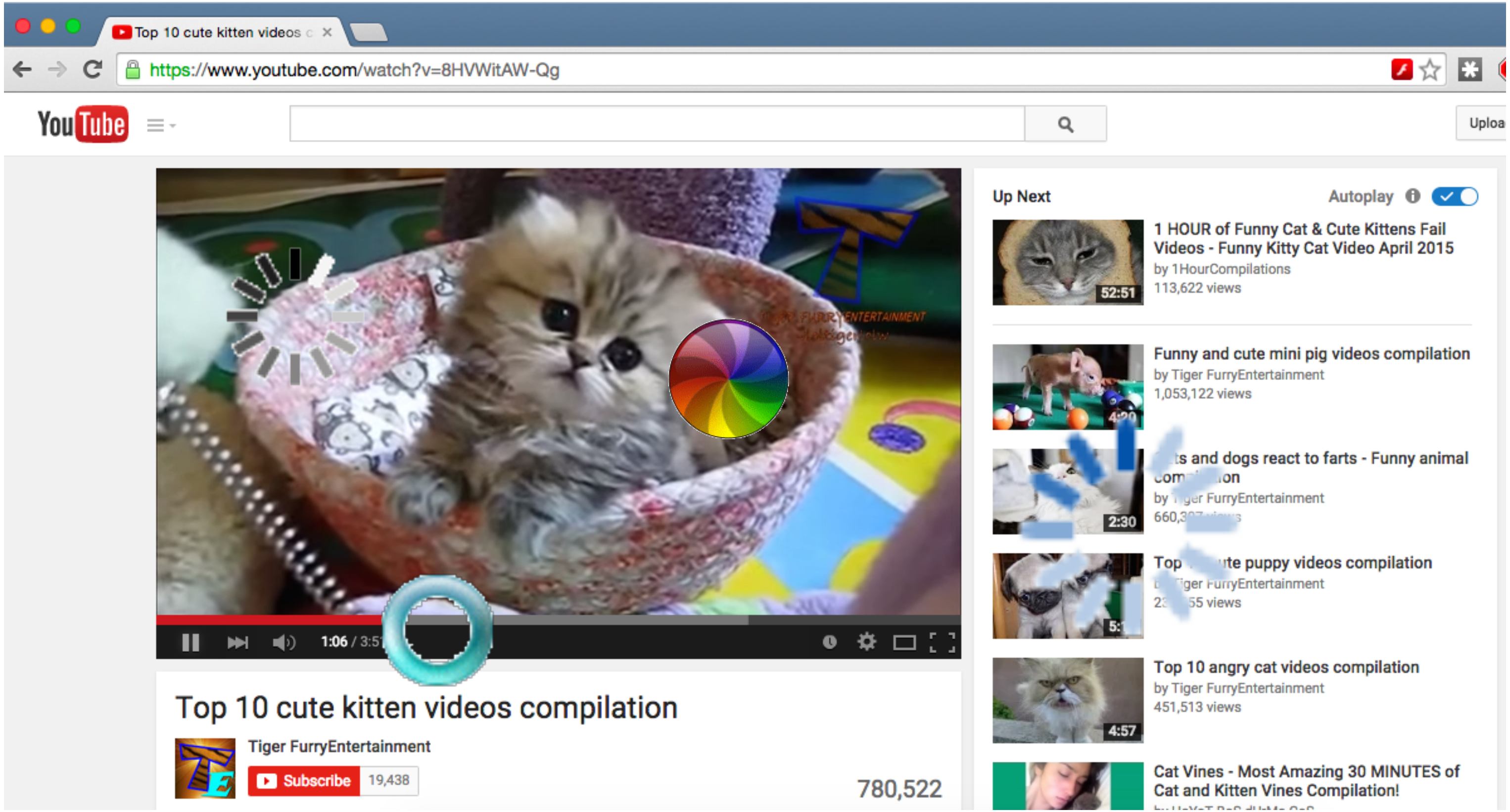
Cats and dogs react to farts - Funny animal compilation
by Tiger FurryEntertainment
660,307 views
2:30

Top 10 cute puppy videos compilation
by Tiger FurryEntertainment
233,055 views
5:11

Top 10 angry cat videos compilation
by Tiger FurryEntertainment
451,513 views
4:57

Cat Vines - Most Amazing 30 MINUTES of Cat and Kitten Vines Compilation!

Motivation



Top 10 cute kitten videos c

https://www.youtube.com/watch?v=8HVWitAW-Qg

Up Next

1 HOUR of Funny Cat & Cute Kittens Fail Videos - Funny Kitty Cat Video April 2015
by 1HourCompilations
113,622 views

Funny and cute mini pig videos compilation
by Tiger FurryEntertainment
1,053,122 views

ts and dogs react to farts - Funny animal com ion
by ger FurryEntertainment
660,3

Top te puppy videos compilation
iger FurryEntertainment
2 55 views

Top 10 angry cat videos compilation
by Tiger FurryEntertainment
451,513 views

Cat Vines - Most Amazing 30 MINUTES of Cat and Kitten Vines Compilation!

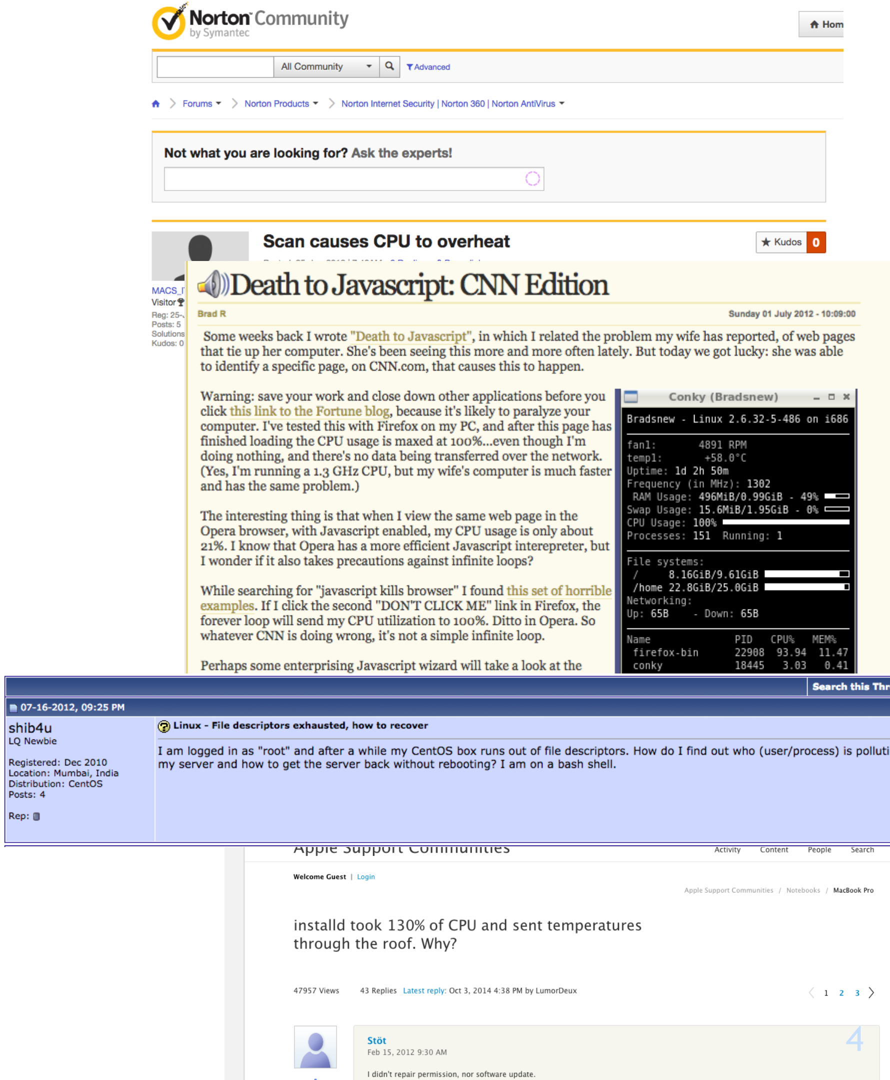
Top 10 cute kitten videos compilation

Tiger FurryEntertainment

780,522

Typical Causes

- DropBox sync
- Browser tabs
- Virus scan
- Software updates
- …



Not what you are looking for? Ask the experts!

Scan causes CPU to overheat

Death to Javascript: CNN Edition

Brad R — Sunday 01 July 2012 - 10:09:00

Some weeks back I wrote "Death to Javascript", in which I related the problem my wife has reported, of web pages that tie up her computer. She's been seeing this more and more often lately. But today we got lucky: she was able to identify a specific page, on CNN.com, that causes this to happen.

Warning: save your work and close down other applications before you click this link to the Fortune blog, because it's likely to paralyze your computer. I've tested this with Firefox on my PC, and after this page has finished loading the CPU usage is maxed at 100%...even though I'm doing nothing, and there's no data being transferred over the network. (Yes, I'm running a 1.3 GHz CPU, but my wife's computer is much faster and has the same problem.)

The interesting thing is that when I view the same web page in the Opera browser, with Javascript enabled, my CPU usage is only about 21%. I know that Opera has a more efficient Javascript intereprеter, but I wonder if it also takes precautions against infinite loops?

While searching for "javascript kills browser" I found this set of horrible examples. If I click the second "DON'T CLICK ME" link in Firefox, the forever loop will send my CPU utilization to 100%. Ditto in Opera. So whatever CNN is doing wrong, it's not a simple infinite loop.

Perhaps some enterprising Javascript wizard will take a look at the

Conky (Bradsnew)

```
Bradsnew - Linux 2.6.32-5-486 on i686
fan1:       4891 RPM
temp1:       +58.0°C
Uptime: 1d 2h 50m
Frequency (in MHz): 1302
 RAM Usage: 496MiB/0.99GiB - 49%
Swap Usage: 15.6MiB/1.95GiB - 0%
CPU Usage: 100%
Processes: 151  Running: 1

File systems:
 /     8.16GiB/9.61GiB
 /home 22.8GiB/25.0GiB
Networking:
Up: 65B    - Down: 65B

Name              PID   CPU%   MEM%
 firefox-bin      22908  93.94  11.47
 conky            18445   3.03   0.41
```

07-16-2012, 09:25 PM

shib4u
LQ Newbie

Registered: Dec 2010
Location: Mumbai, India
Distribution: CentOS
Posts: 4

Rep:

Linux - File descriptors exhausted, how to recover

I am logged in as "root" and after a while my CentOS box runs out of file descriptors. How do I find out who (user/process) is polluti my server and how to get the server back without rebooting? I am on a bash shell.

Apple Support Communities
Activity
Content
People
Search
Welcome Guest | Login
Apple Support Communities / Notebooks / MacBook Pro

installd took 130% of CPU and sent temperatures through the roof. Why?

47957 Views 43 Replies Latest reply: Oct 3, 2014 4:38 PM by LumorDeux

Stöt
Feb 15, 2012 9:30 AM

I didn't repair permission, nor software update.

Goal: Performance Isolation

- Control performance, battery, heat, etc.
  - Do not kill -- useful-but-gluttonous apps
- Do not require OS / hardware changes
  - Applicable to sandboxes, browsers, etc.
  - Run everywhere (Linux, Windows, Mac, Android, OpenWrt, etc.)
- Focus on mechanism
  - Necessary for policies to function

Why is performance isolation hard?

- Multiple contended resources
  - Separate mechanisms / policies
  - Creating an overarching policy is difficult

- Some controls are missing
  - Gaps in enforcement -> lack of isolation

- Legacy systems tend to be work-preserving

Key Idea: Uniform Resource Control

- Unifying resource abstraction
    - Two axes per resource: fungible/renewable
        - Fully defines mechanism
    - Easy to cover new resources
    - Easy to implement policies

Resource Abstraction Questions

- Fungible:
  - Are items of this type interchangeable?
    - Yes (disk space) vs No (TCP port)

- Renewable:
  - Are items replenished over time?
    - Yes (Network bandwidth) vs No (RAM)

Resource Controls

	Not Fungible	Fungible

Not Renewable	`is_item_allowed()`
	“Check if permitted”
	e.g. UDP port
		`tattle_add_item()`
		`tattle_remove_item()`
		“Restrict total used”
		e.g. File Descriptors

Renewable	`tattle_quantity()` “Rate limit” e.g. Network b/w	`tattle_quantity()` “Rate limit” e.g. CPU

Enforcement mechanism

- Polling
  - Find value, stop / rate limit if over
    - e.g. CPU uses job control interface (`SIGSTOP / SIGCONT`)
- Interposition
  - API code changes to add interposition
- Which depends on implementation

Example Implementation Changes

```
def sendmessage(destip,destport,msg,localip,localport):  # 117 lines
  ...



  # get the OS's UDP socket
  sock = _get_udp_socket(localip, localport)


  # Send this UDP datagram
  bytessent = sock.sendto(msg, (destip, destport))




  ...
```

Example Implementation Changes

```
def sendmessage(destip,destport,msg,localip,localport):  # 117 lines + 10 lines
  ...
  # check that we are permitted to use this port...
  if not fence.is_item_allowed('UDPport',localport):
    raise ResourceAccessDenied('...')
  # get the OS's UDP socket
  sock = _get_udp_socket(localip, localport)


  # Send this UDP datagram
  bytessent = sock.sendto(msg, (destip, destport))



  ...
```

UDP port: Non-fungible, non-renewable

Example Implementation Changes

```
def sendmessage(destip,destport,msg,localip,localport):  # 117 lines + 10 lines
  ...
  # check that we are permitted to use this port...
  if not fence.is_item_allowed('UDPport',localport):
    raise ResourceAccessDenied('...')
  # get the OS's UDP socket
  sock = _get_udp_socket(localip, localport)
  # Register this socket descriptor with fence
  fence.tattle_add_item('outsocketsopened', id(sock))
  # Send this UDP datagram
  bytessent = sock.sendto(msg, (destip, destport))



  ...
```

socket: Fungible, non-renewable

Example Implementation Changes

```
def sendmessage(destip,destport,msg,localip,localport):  # 117 lines + 10 lines
  ...
  # check that we are permitted to use this port...
  if not fence.is_item_allowed('UDPport',localport):
    raise ResourceAccessDenied('...')
  # get the OS's UDP socket
  sock = _get_udp_socket(localip, localport)
  # Register this socket descriptor with fence
  fence.tattle_add_item('outsocketsopened', id(sock))
  # Send this UDP datagram
  bytessent = sock.sendto(msg, (destip, destport))
  # Account for the network bandwidth utilized
  if _is_loopback_ipaddr(destip):
    fence.tattle_quantity('loopbacksend', bytessent + 64)
  else:
    fence.tattle_quantity('internetsend', bytessent + 64)
  ...
```

Network b/w: Fungible, Renewable

Uses of Fence

- Seattle Testbed's Repy sandbox
  - Seattle ~= Peer-to-peer PlanetLab
  - Tens of thousands of diverse devices
- Lind
  - NaCl / POSIX sandbox
- Sensibility Testbed
  - Privacy preserving sensing on Android

Limitations

- Resource consumption must be visible
  - HW / OS hide info
- Minimizes performance impact
  - “Worst case” limits
- Scope of policies
  - Unclear how complete Fence is
  - Worked for us in practice

Evaluation

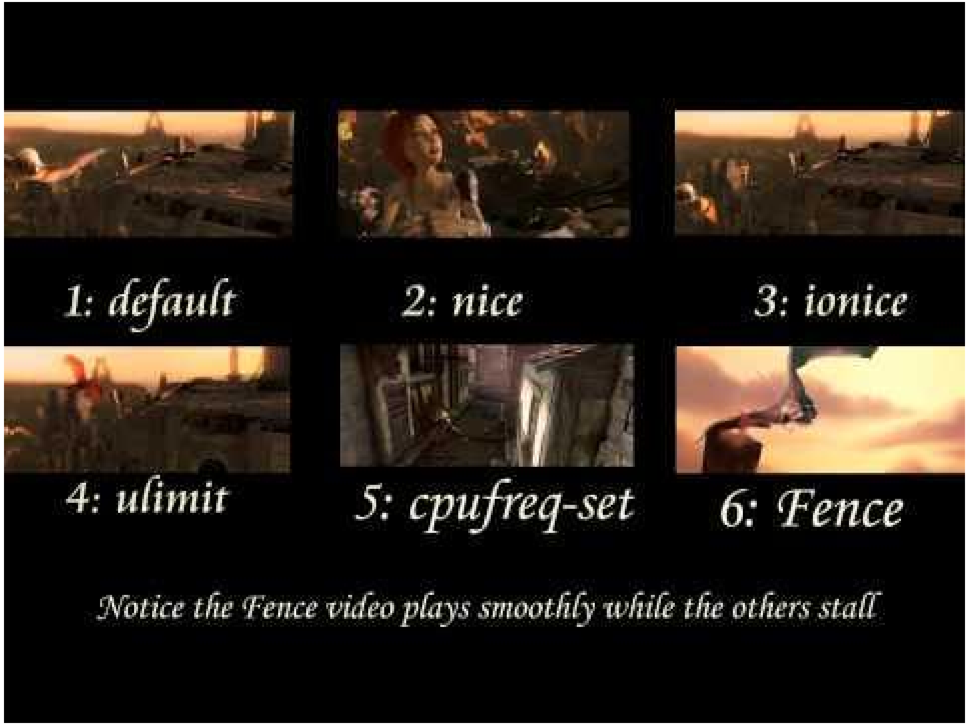
- How well does Fence work vs legacy controls?
- How well does Fence work across platforms?
- How much overhead does Fence incur?
- Can realistic policies be expressed in Fence?
- How diverse of resources can be metered?
- How hard is it to add resources to Fence?

Evaluation

- How well does Fence work vs legacy controls?
- How well does Fence work across platforms?
- How much overhead does Fence incur?
- Can realistic policies be expressed in Fence?
- How diverse of resources can be metered?
- How hard is it to add resources to Fence?

Fence vs Legacy Controls

- Video on disk (Dell Inspirion 630m w/ Ubuntu 10.04)
- “hog” everything
- worst setting for hog
- best setting for video



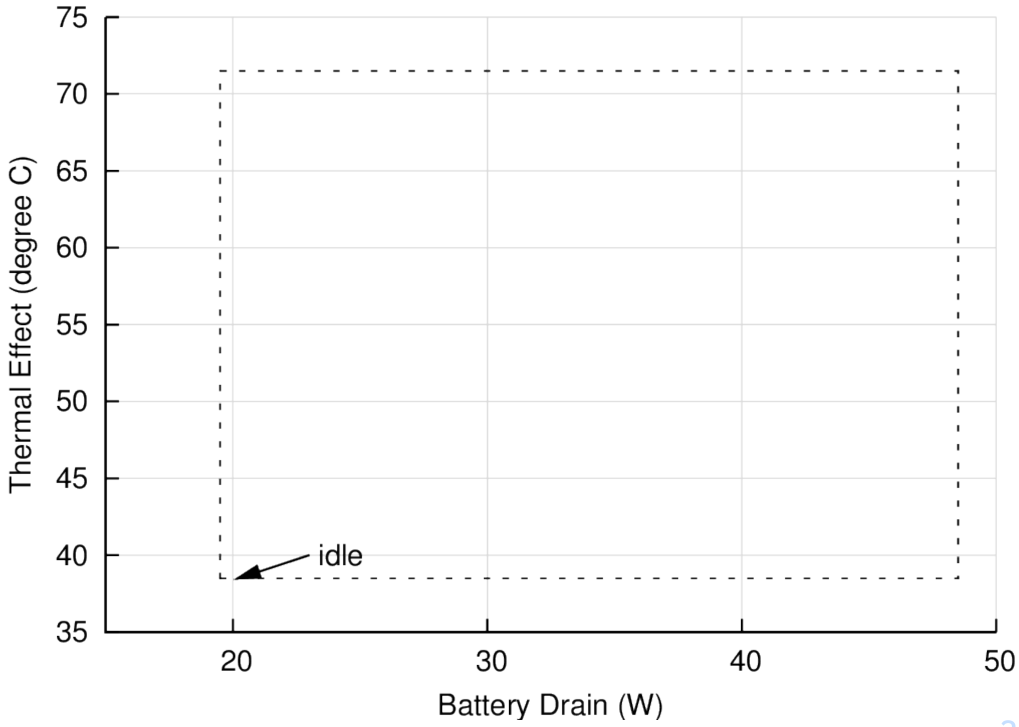
Notice the Fence video plays smoothly while the others stall

https://www.youtube.com/watch?v=OGe8QpPbtz4

Fence vs Legacy Heat / Battery

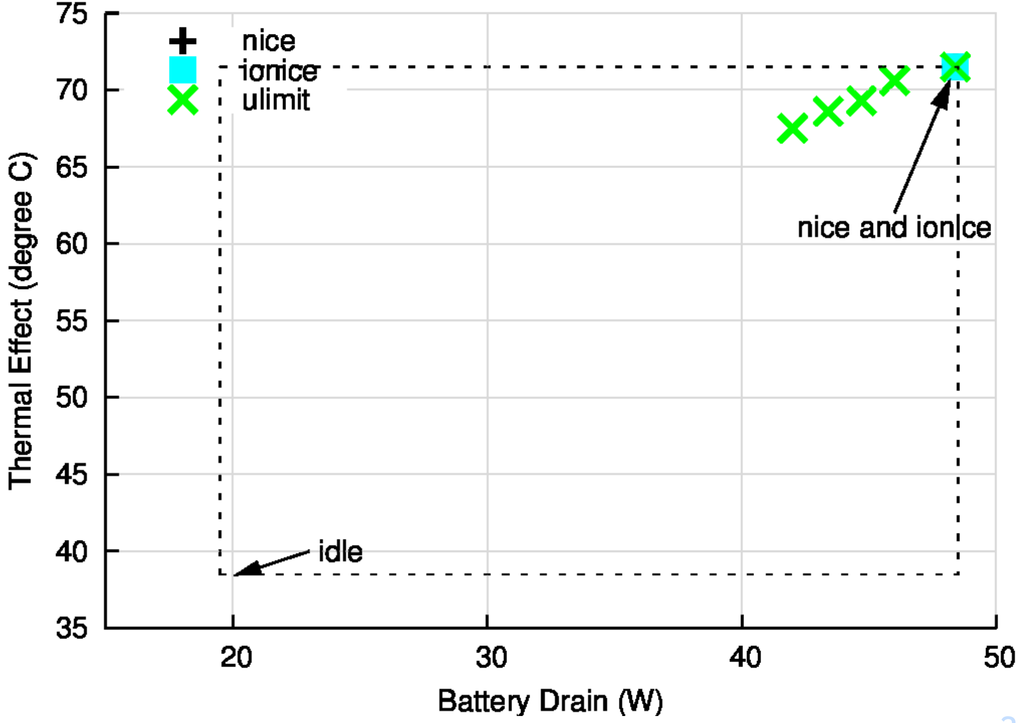
- Heat / battery

- “hog” everything



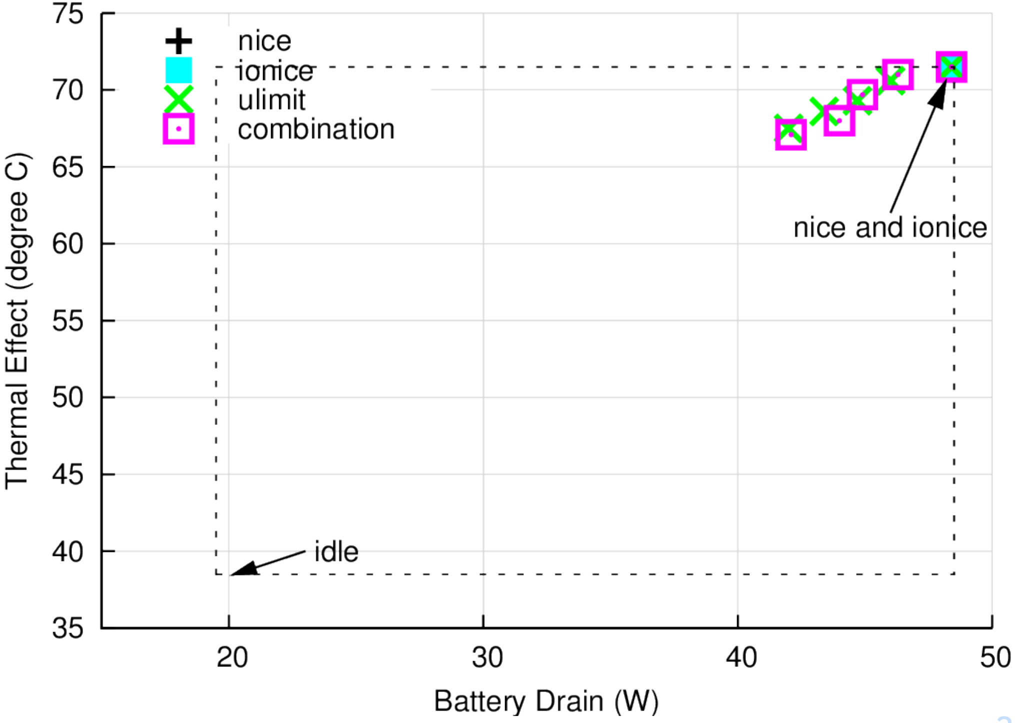
Fence vs Legacy Heat / Battery

- Heat / battery
- “hog” everything



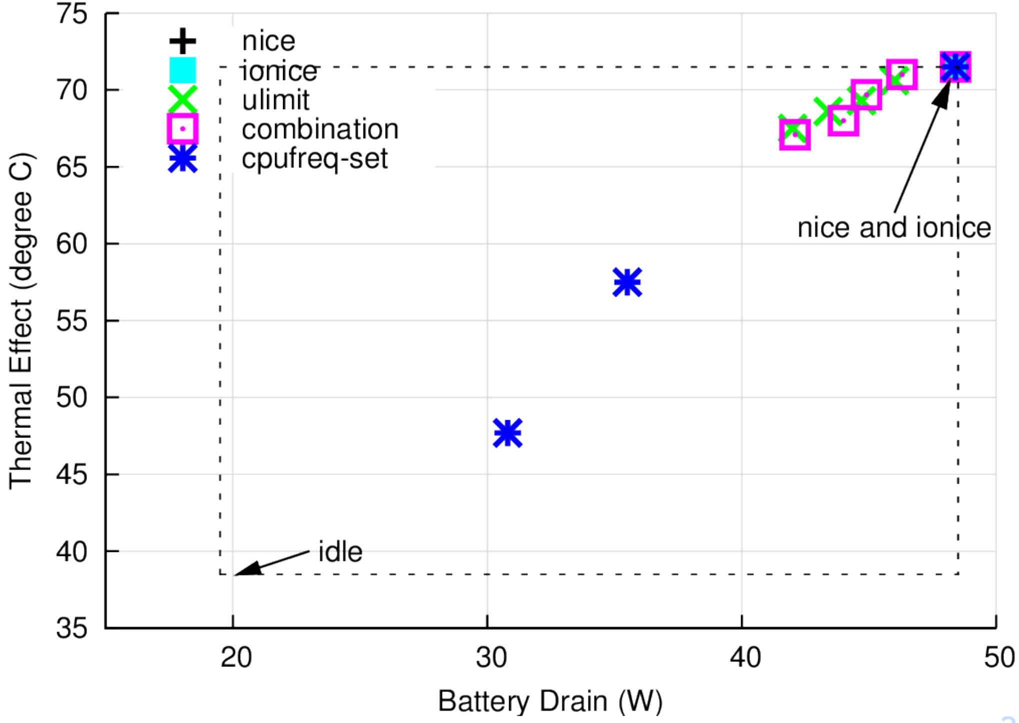
Fence vs Legacy Heat / Battery

- Heat / battery
- “hog” everything



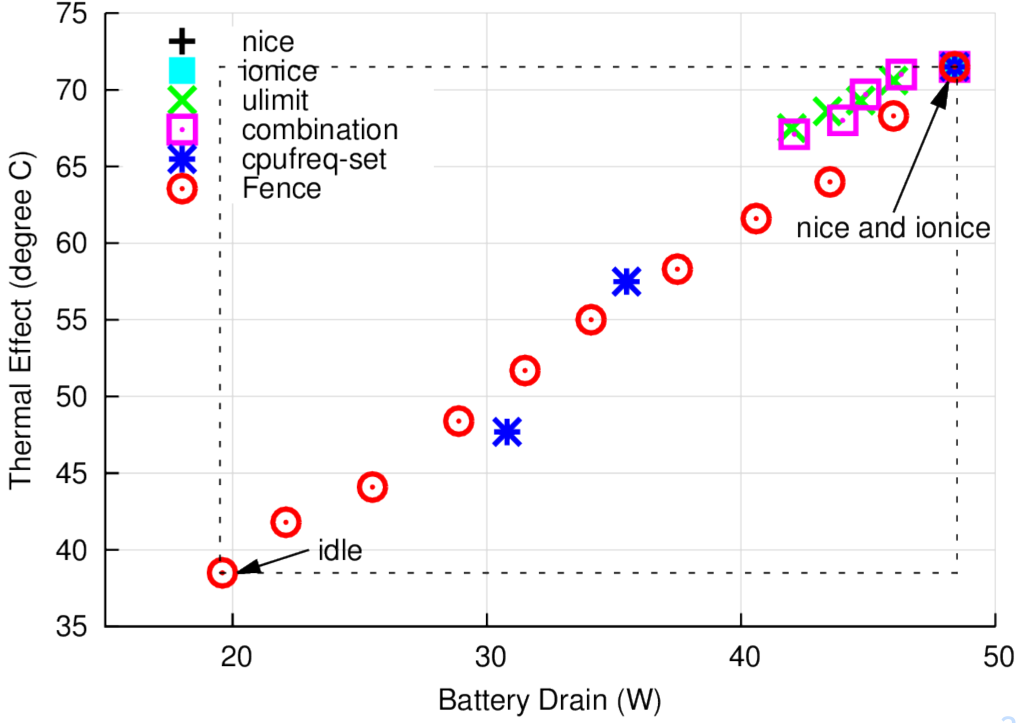
Fence vs Legacy Heat / Battery

- Heat / battery
- “hog” everything



Fence vs Legacy Heat / Battery

- Heat / battery
- “hog” everything



Evaluation

- ~~How well does Fence work vs legacy controls?~~
- How well does Fence work across platforms?
- How much overhead does Fence incur?
- Can realistic policies be expressed in Fence?
- How diverse of resources can be metered?
- How hard is it to add resources to Fence?

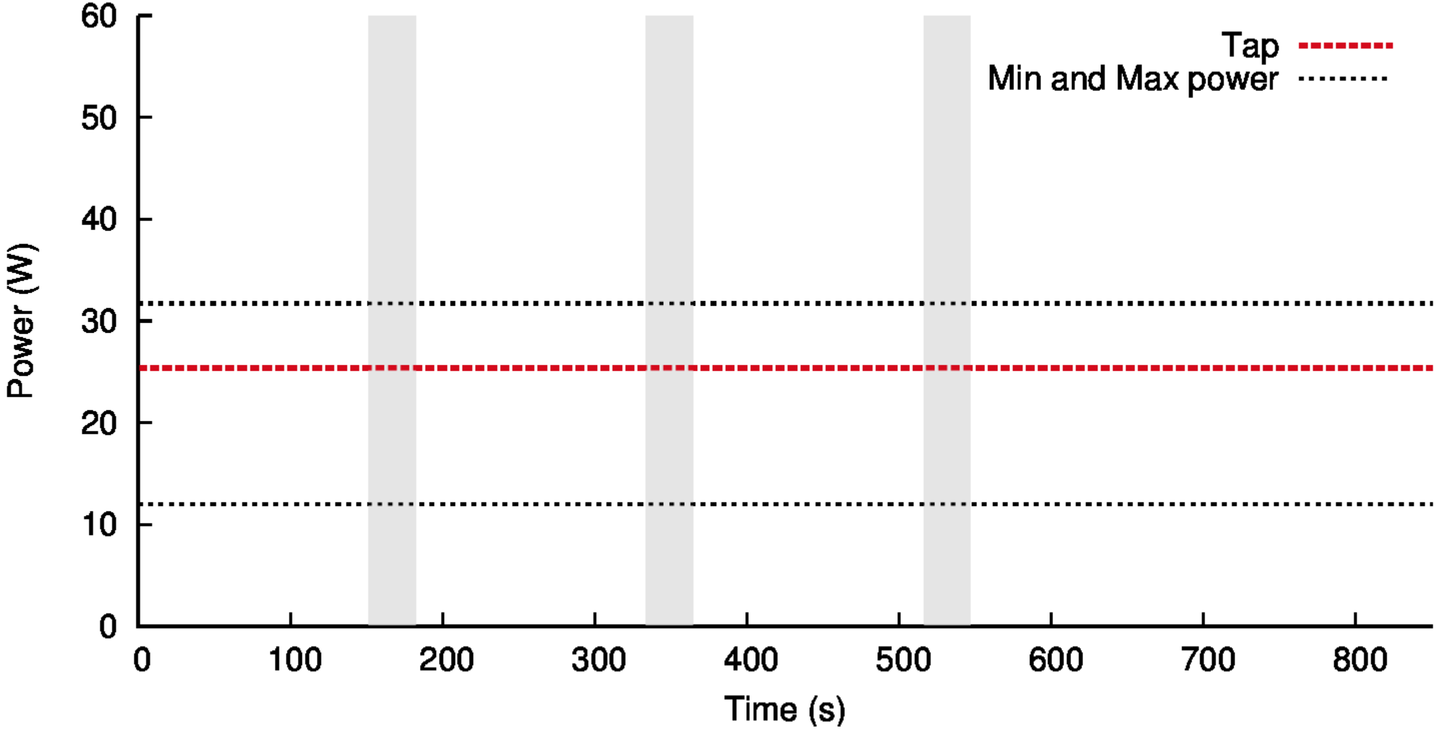
Expressing Policies: Cinder

Power draw policy from Cinder [Roy Eurosys 2011]

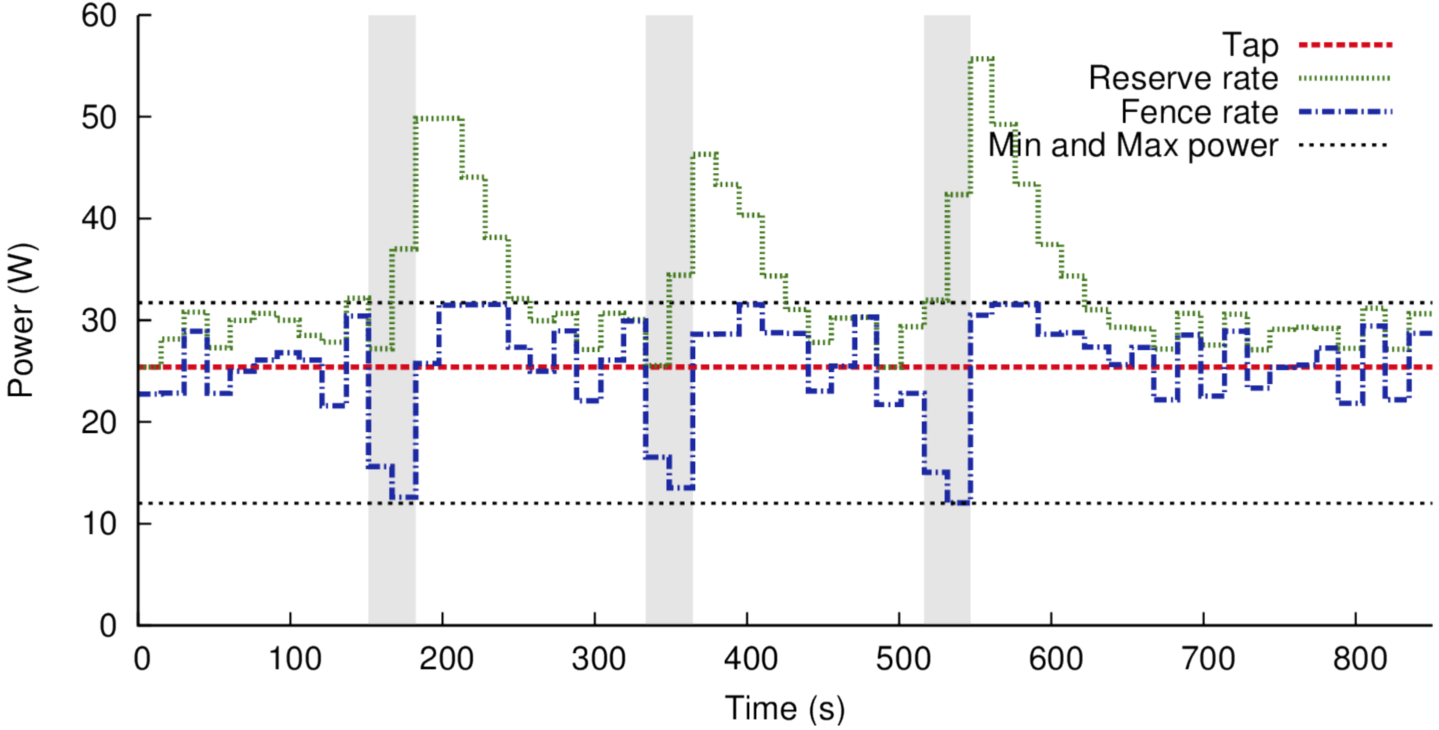
- Stores energy w/ a tap (token bucket)
- Polling using ACPI (updates every 15 seconds)

Program: Richards benchmark in a run / sleep cycle

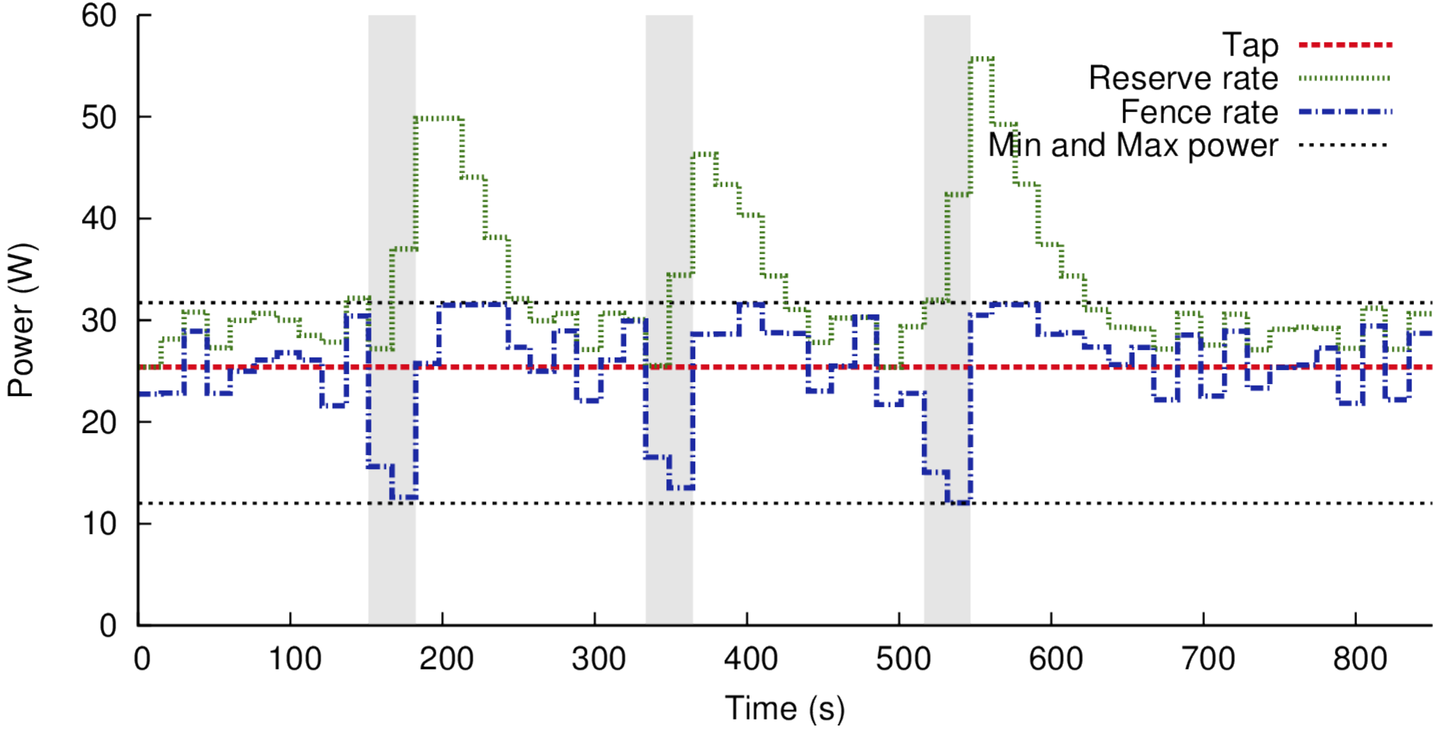
Expressing Policies: Cinder (cont)



Expressing Policies: Cinder (cont)



Expressing Policies: Cinder (cont)



150 LOC!

Conclusion

- Performance isolation is still a challenge

- Uniform Resource Control
    - Same simple reasoning for all resources
    - Fungible / Renewable
    - Easy to implement / use
    - Effective in practice

NYU is Hiring!

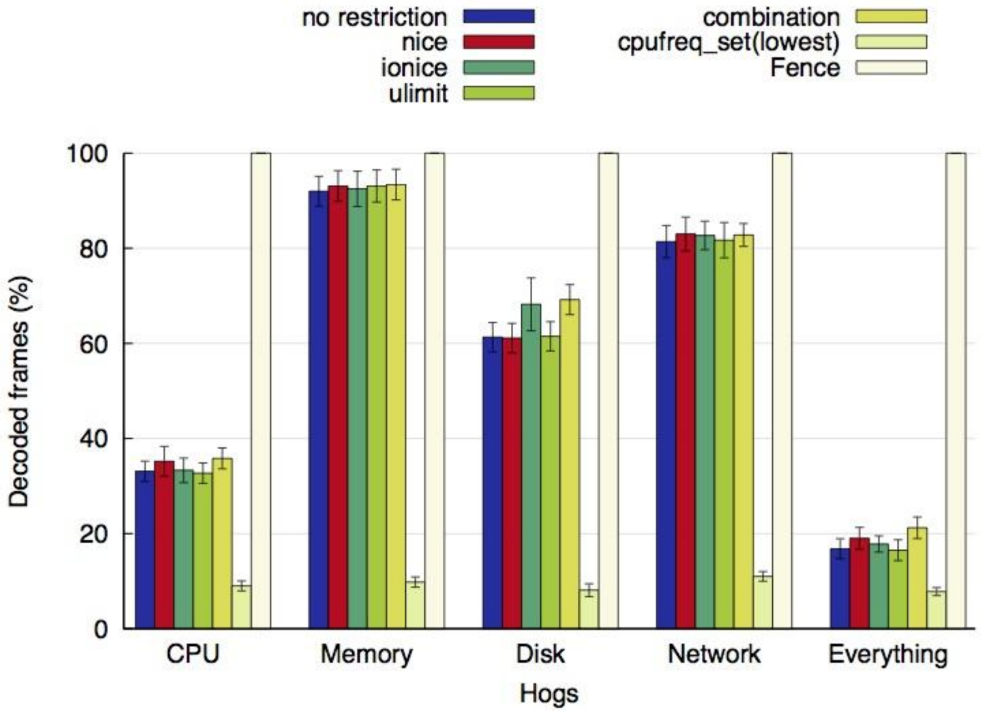
- Secure software distribution
  - Adoption by Python, Ruby, Docker, LEAP, CoreOS, Go, Rust, Haskell, OCaml, etc.
  - Plausible standard for many new domains
- Hiring Post Doc / Research Professor / Dev

Questions?

Fence vs Legacy Controls (cont)

<20% frames for each legacy tool (Combining tools, only gives 22% of frames)

99% of frames for Fence



Example Resource Categorization

	Not Fungible	Fungible
Not Renewable	UDP ports TCP ports	Threads Memory (RAM) Storage Space Open Sockets Open Files
Renewable	Network read / write	CPU File read / write HW random