

# **Power Management for Networked Systems**

Sylvia Ratnasamy (Intel Research Berkeley)

Work in collaboration with  
UC Berkeley, Univ. of Washington and  
Lawrence Berkeley National Lab

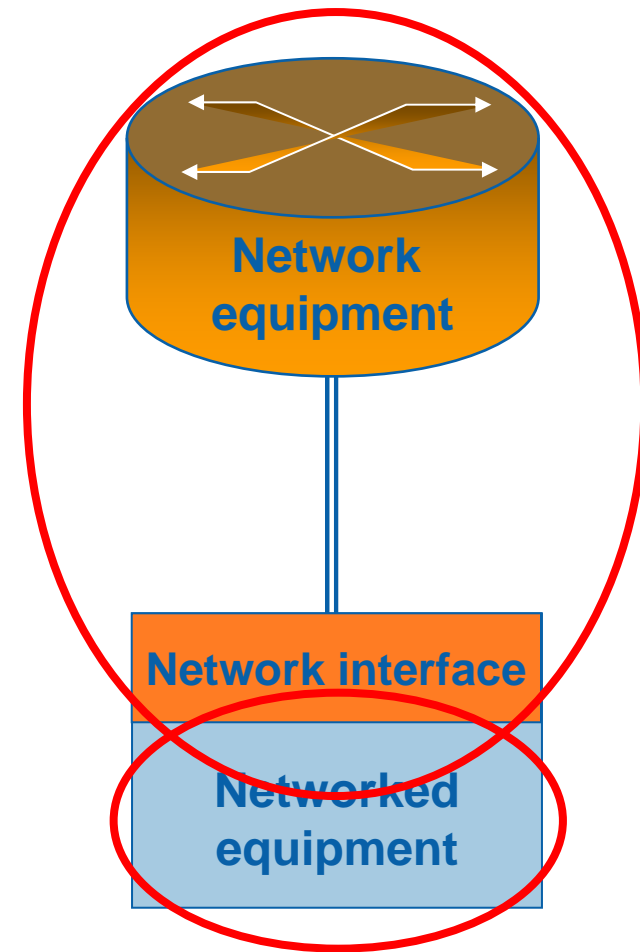
# How do networks contribute to energy use?

**Direct:** consumed in network equipment

- routers, switches, NICs, ...

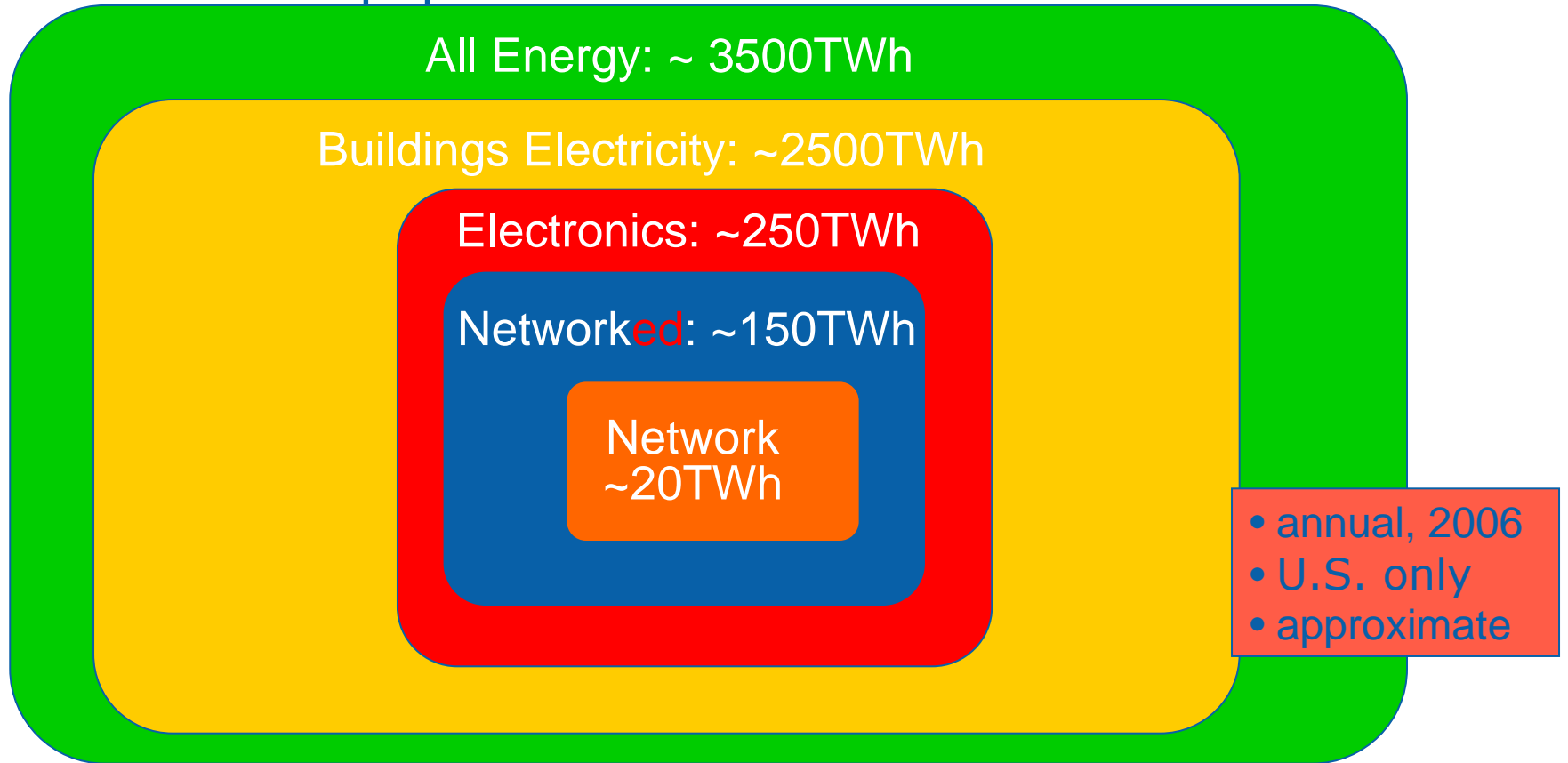
**Indirect:** induced in networked equipment

- PCs, printers, set-top boxes, IP phones, ...
- often left on for remote access



# Our focus

Reducing the energy consumption due to network and networked equipment



# Our focus

Reducing the energy consumption due to network and networked equipment

Represents only <5% of overall use, but...

- still a very large number (20TWh: ~\$2B, 2 nuclear reactor units)
- and growing rapidly
  - more devices being networked
  - bandwidth doubles every 12-18 months
  - 92Tbps Cisco CRS consumes 1MegaWatt!
  - network contributes 25% of data-center energy costs

New efforts: IEEE Energy-Efficient Ethernet group, Cisco Green Sem.

# Opportunity

Utilization is low...

- in **networking equipment:**

Network	Util.
Internet backbones	15 %
Private line networks	3-5%
LANs	1%

- and **networked equipment:**

- office, home PCs typically idle at night
- enterprise laptops < 5%
- google servers run at 10-50% utilization [Barrosso'07]

# Opportunity

Utilization is low...

... but energy consumption when idle is high

- in **networking equipment**:
  - e.g., Cisco GSR line-card draws ~**80W when idle**, ~**90W active**  
[Chabarek,Infocom08]

A lot of network-related energy consumption is wasted!

Goal: consumption that scales with utilization

# Outline

- Reducing the energy consumption of network equipment
  - solutions based on sleep and rate-adaptation
- Reducing the energy consumption of networked equipment
  - using network proxies for idle end-hosts (ongoing)
- Conclusion

# Reducing in-network energy consumption

Goal: reduce consumption without compromising performance

Key Idea: let networks “nap” or slow down when lightly loaded

Rationale: energy consumption:  $E \sim p_{\text{idle}} T_{\text{idle}} + p_{\text{active}} T_{\text{active}}$

Inspiration: sleep and performance states in PCs, processors  
Sleeping reduces first term      Slowing reduces both terms

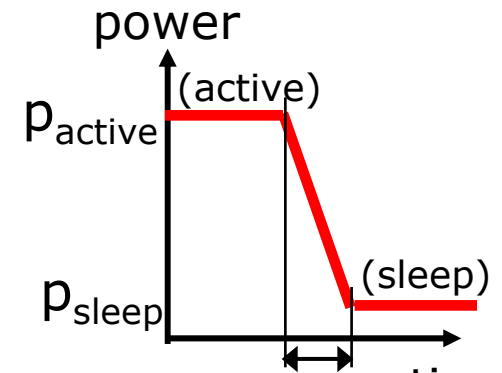
## Challenge:

- appropriate hardware-level primitives      ← model
- higher-layer algorithms that invoke these wisely      ← design, evaluate

# Hardware Support

## Sleep

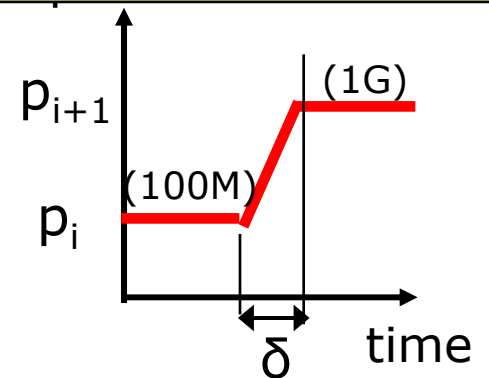
- Single sleep state with  $p_{\text{sleep}} \ll p_{\text{idle}}$
- $\delta$  : transition period (ms)



determines the resultant power savings

## Rate-adaptation

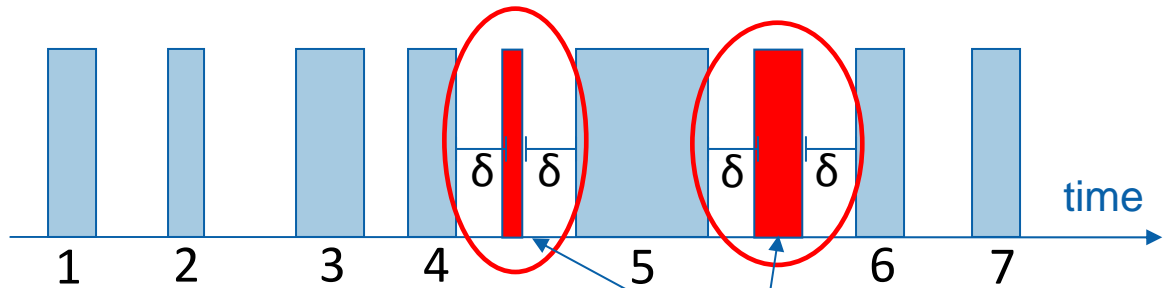
- N rates states
- Rates  $r_1, \dots, r_n$  and  $p_i < p_{i+1}$
- $\delta$  : transition period (ms)



affects how well the network can sleep/rate-adapt

# Sleeping in the network

Packets over a link:

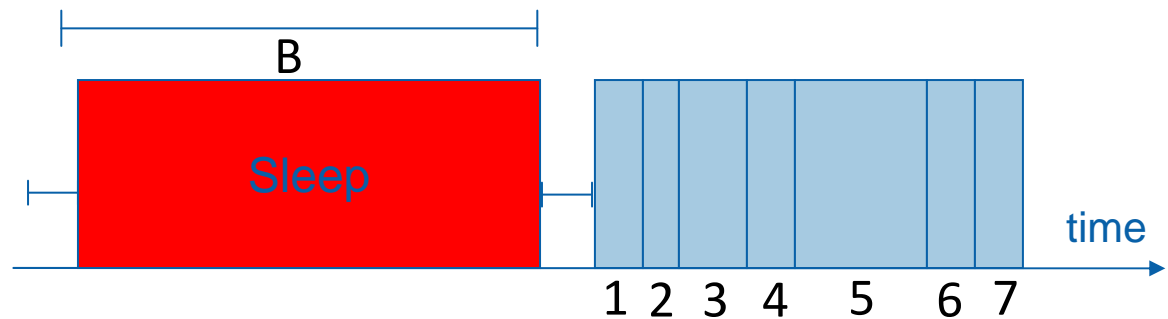


Sleep time depends on:



Buffer and burst (B&B):

- buffer for "**B**" ms
- only at first-hop
- bounds delay

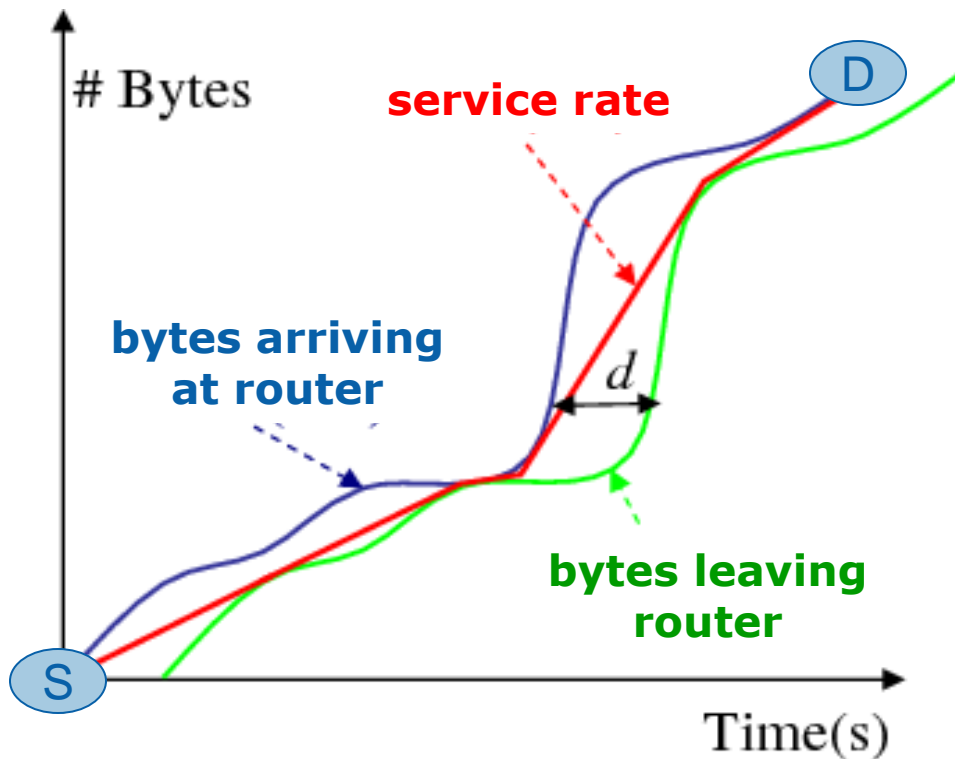


Optimal strategy: first-hop nodes buffer, coordinate transmission times

Practical strategy: first-hop nodes buffer, but do so independently

# Rate adaptation

Basic idea: decrease rate as much as possible without introducing more than  $d$  ms per hop



Optimal algorithm: picks schedule of rates based on complete knowledge of future traffic arrivals.

Practical algorithm: approximates the above using only local information (packet arrival rate, local queue size).

# How effective are sleep & rate adaptation?

## Simulations driven by real network traffic, topologies

- Abilene backbone
- Intel corporate network ( $\sim 400$  routers)

## Results using practical algorithms, $\delta < 1\text{ms}$

- can sleep for  $\sim 70\%$  of the total idle time
- can run links at  $< 10\text{--}20\%$  of actual utilization (1/2.5/5/10Gbps)
- performance is within 15% of optimal

## Performance penalty

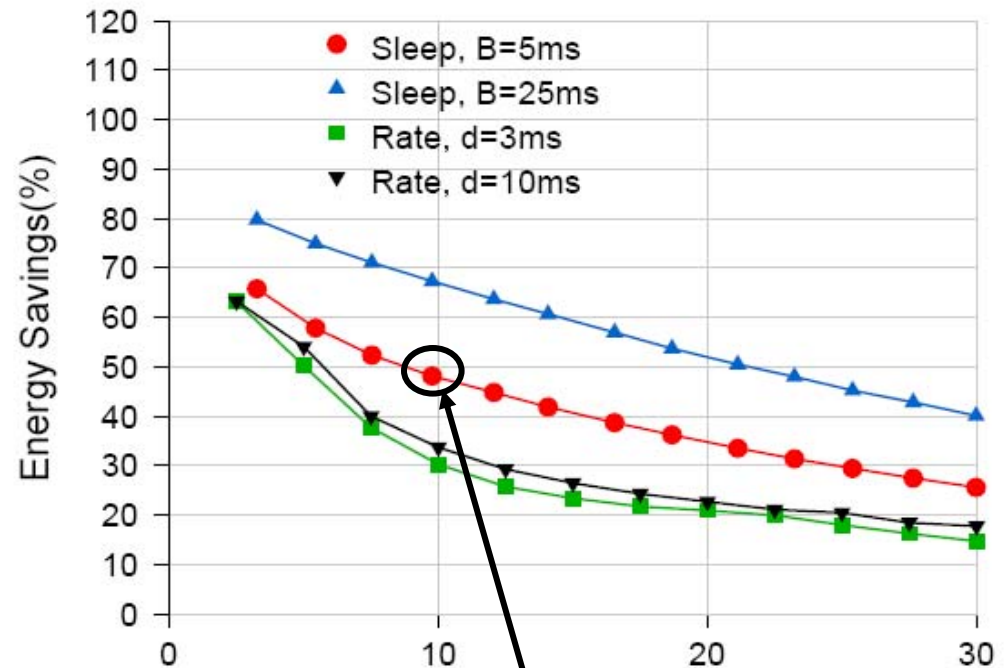
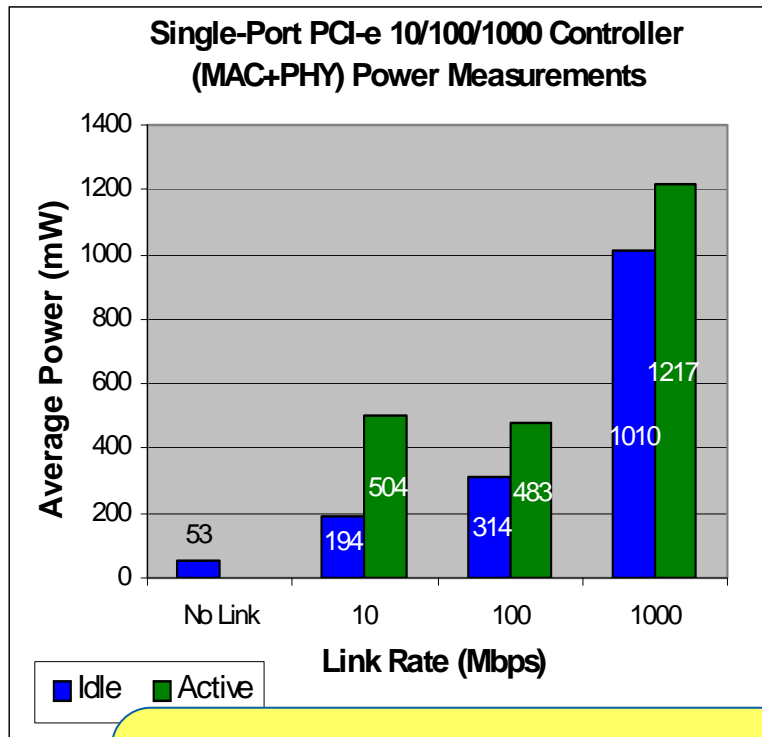
- $< 10\text{ms}$  additional end-to-end delay (i.e.,  $B < 10\text{ms}$ ,  $d \sim 3\text{ms}$ )
- no additional packet loss

Benefits  $\rightarrow$  energy savings? Depends on hardware  $p_{\text{sleep}}$ ,  $p_{\text{rate}}$

# Current Hardware: e.g., Intel NIC

(similar findings for a Cisco GSR)

Abilene backbone; transition time=1ms; rates=1G/100M/10Mbps

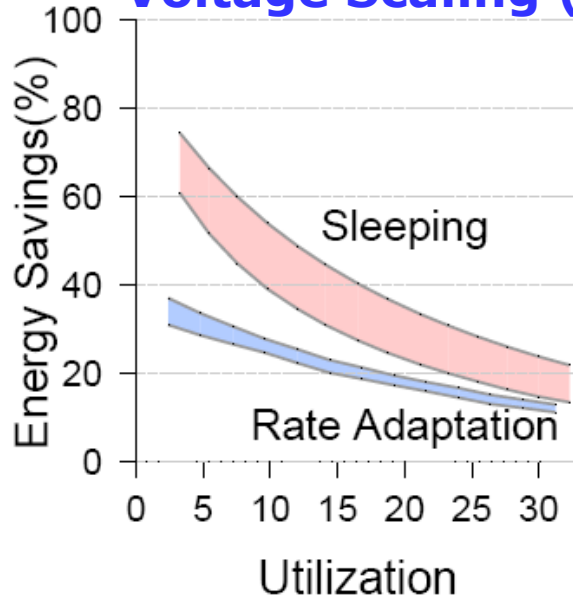


Sleep-based techniques based on B&B are currently being standardized by the 802.3az Energy Efficient Ethernet Taskforce



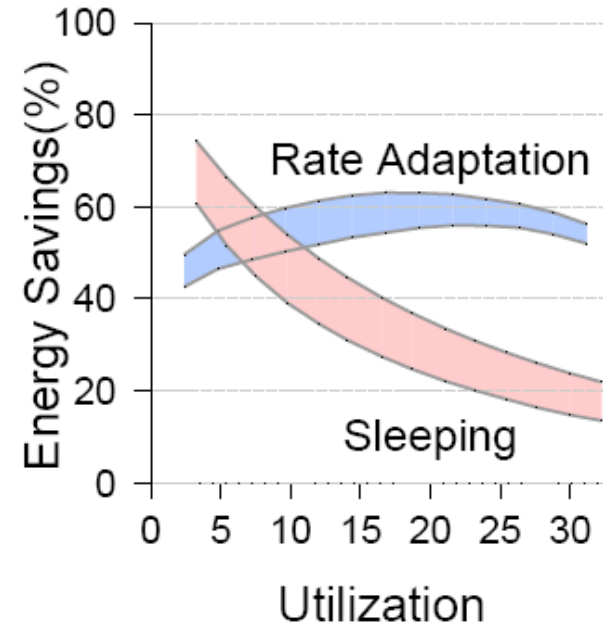
# Future Improved Hardware?

## Without Dynamic Voltage Scaling (DVS)



(a)  $C = 0.1, \beta = 0.1$

## With DVS, 10 rates, $\delta=0.1$ to 1ms



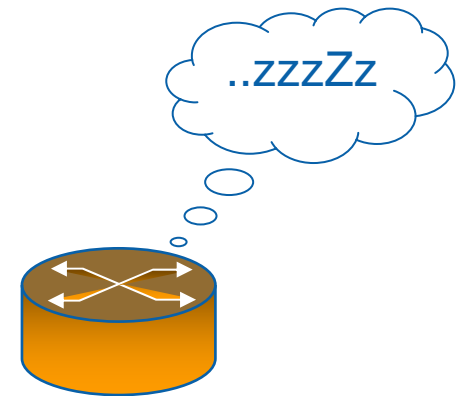
(a)  $C = 0.1, \beta = 0.1$

Nature of hardware support qualitatively impacts the sleeping vs. rate-adaptation question

# Summary: in-network energy

Much potential value in sleep/rate states for network equipment:

- stand to halve energy consumption for lightly loaded links (10-20%)
- can do so with bounded impact on performance (small added delay)
- can realize significant gains with simple, practical schemes



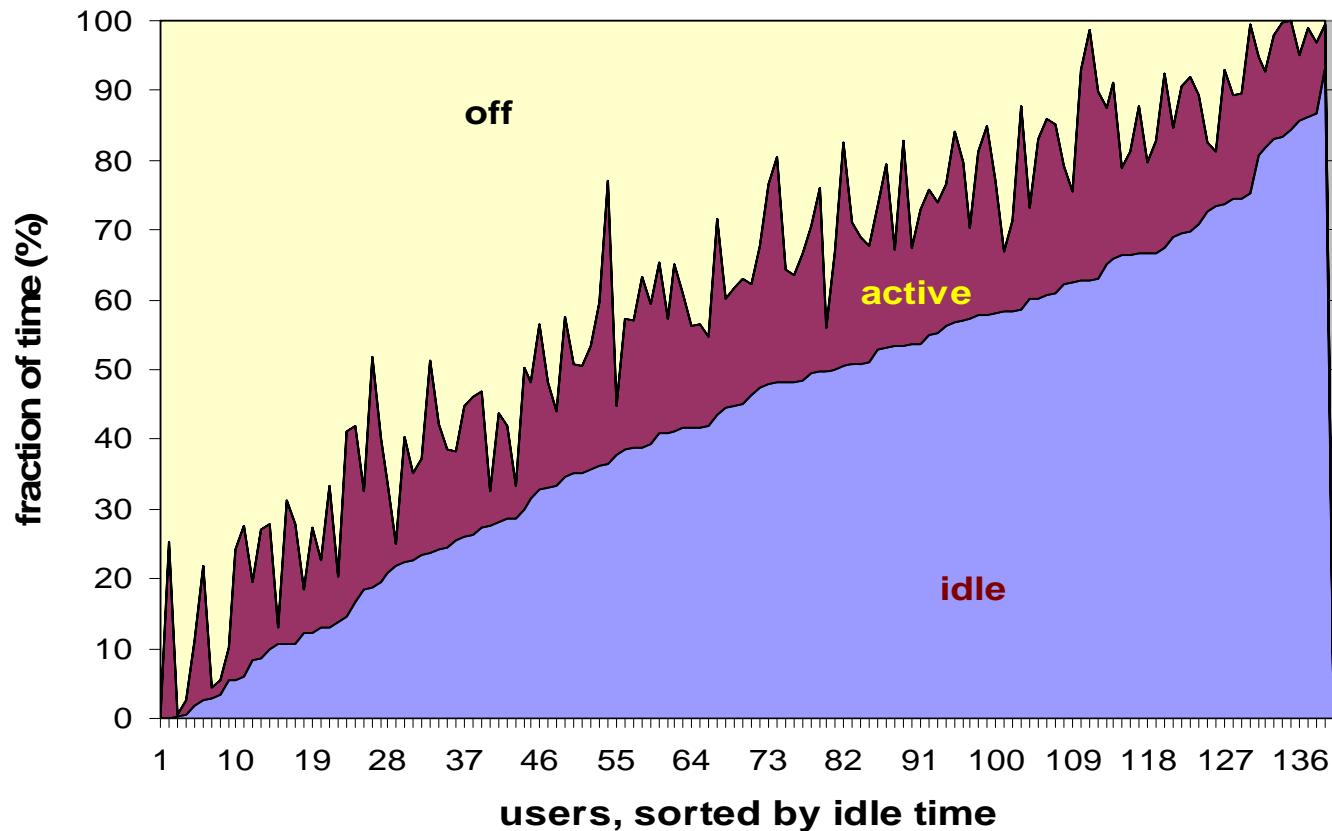
More details and results in "Reducing Network Energy Consumption via Rate-adaptation and Sleeping," Nedevschi, et. al., Usenix NSDI'08

# Outline

- Reducing the energy consumption of network equipment
  - solutions based on sleep and rate-adaptation
- Reducing the energy consumption of networked equipment
  - using network proxies for idle end-hosts (ongoing)
- Conclusion

# The problem

- End-systems are often idle but their idle-time power draw remains high
  - 140W when idle vs. 153W when active

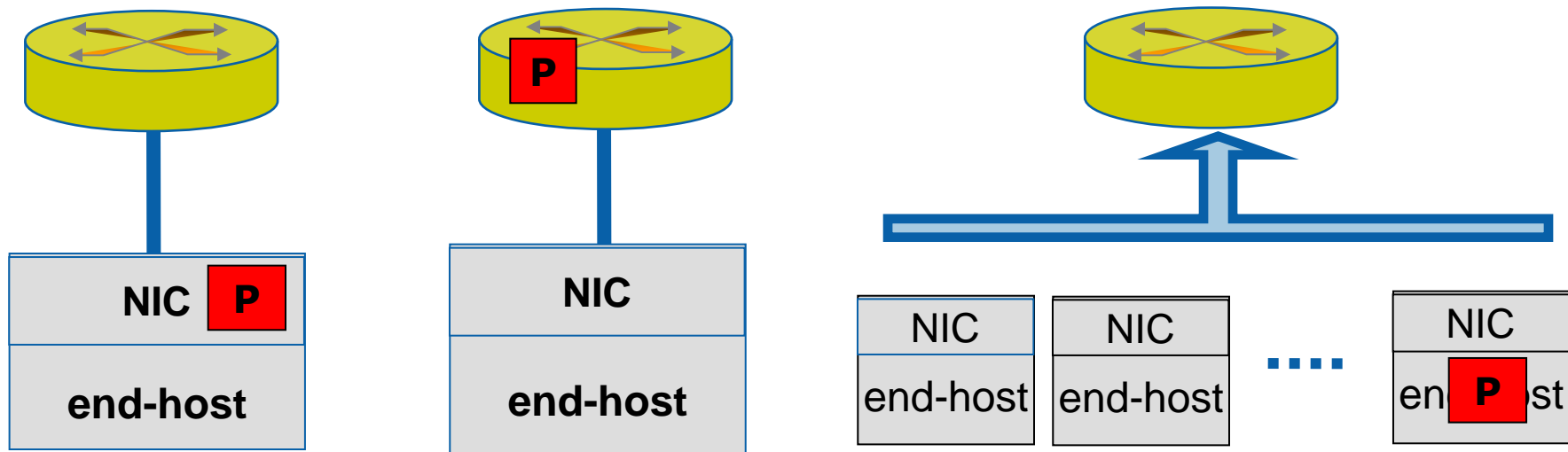


# The problem

- End-systems are often idle but their idle-time power draw remains high
  - 140W when idle vs. 153W when active
- One option: put system into sleep mode
  - only draws 1-2W
  - problem: lose network presence
- An old solution: “magic” packet, wakes up a sleeping host
  - problem: “chatty” protocols prevents sleep

# Proposed solution: a network proxy

- handles idle-time traffic on behalf of a sleeping host
- multiple deployment options



# Early findings

Evaluate two sample proxies:

- (1) proxy several protocols (ARP, LLC, NBNS, ICMP, IGMP, UPNP, DNS, SNMP, CUPS), wake up for everything else
  - can sleep ~55% of idle time
  - proxying is transparent, but more complex
  
- (2) proxy very few protocols (ARP, NetBios), wake-up only for new TCP connections, drop everything else
  - can sleep ~90% of idle time
  - simple but not transparent

# Conclusions

## Network energy consumption:

- growing issue in an increasingly networked world
- significant opportunity due to low utilizations
- but need a system/network-wide perspective...
- ... need metrics: tradeoff energy savings vs. performance
- ... and need to consider the interdependence of software, hardware and protocols