# Read-Copy-Update

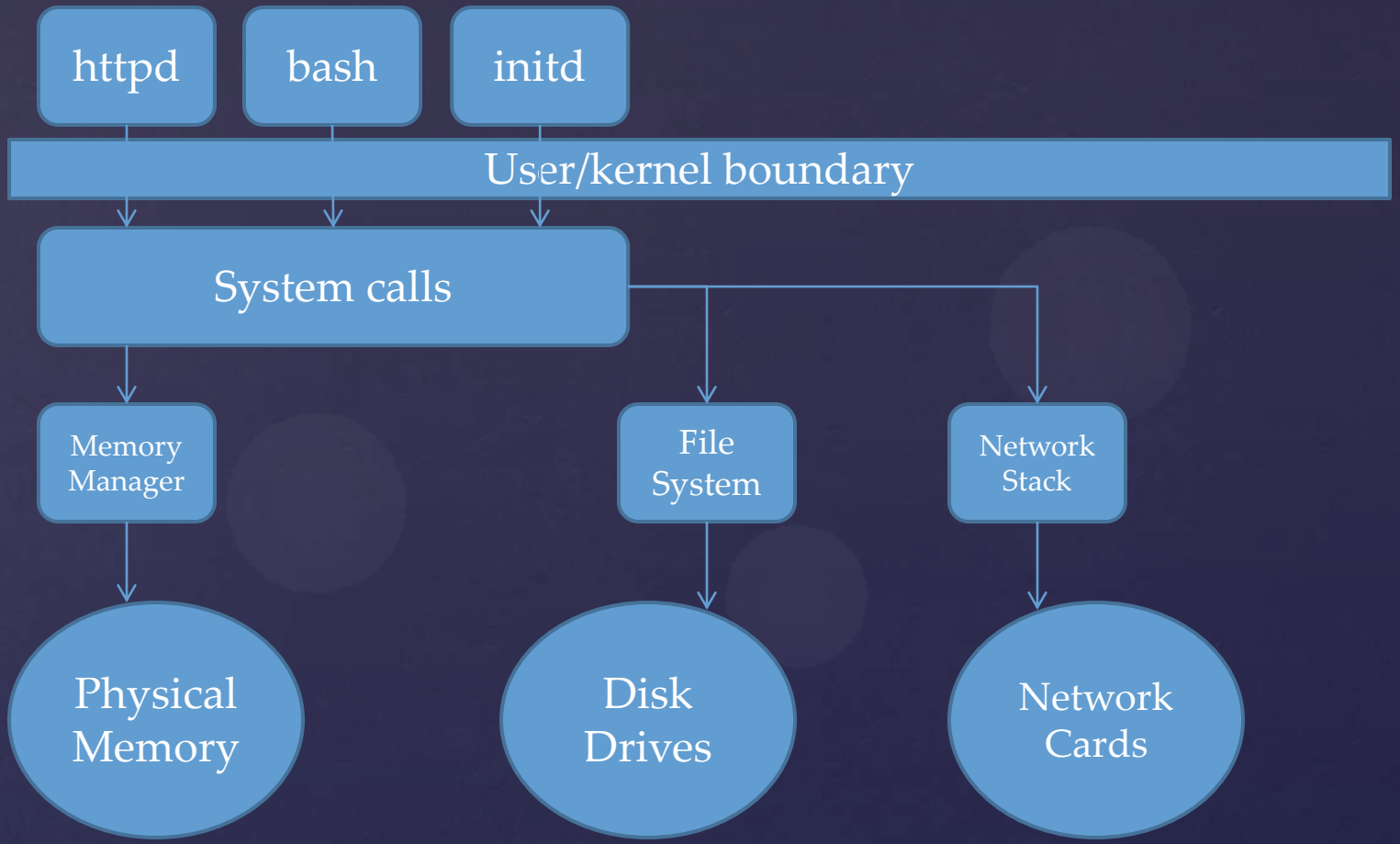
{ Wait-free synchronization for the kernel

# Intro

- What is the kernel
- What is synchronization
- Why is synchronization important
- What is read-copy-update (RCU) and how does it work
- Why is this important

# What is the kernel?

{




# An arbiter of resources

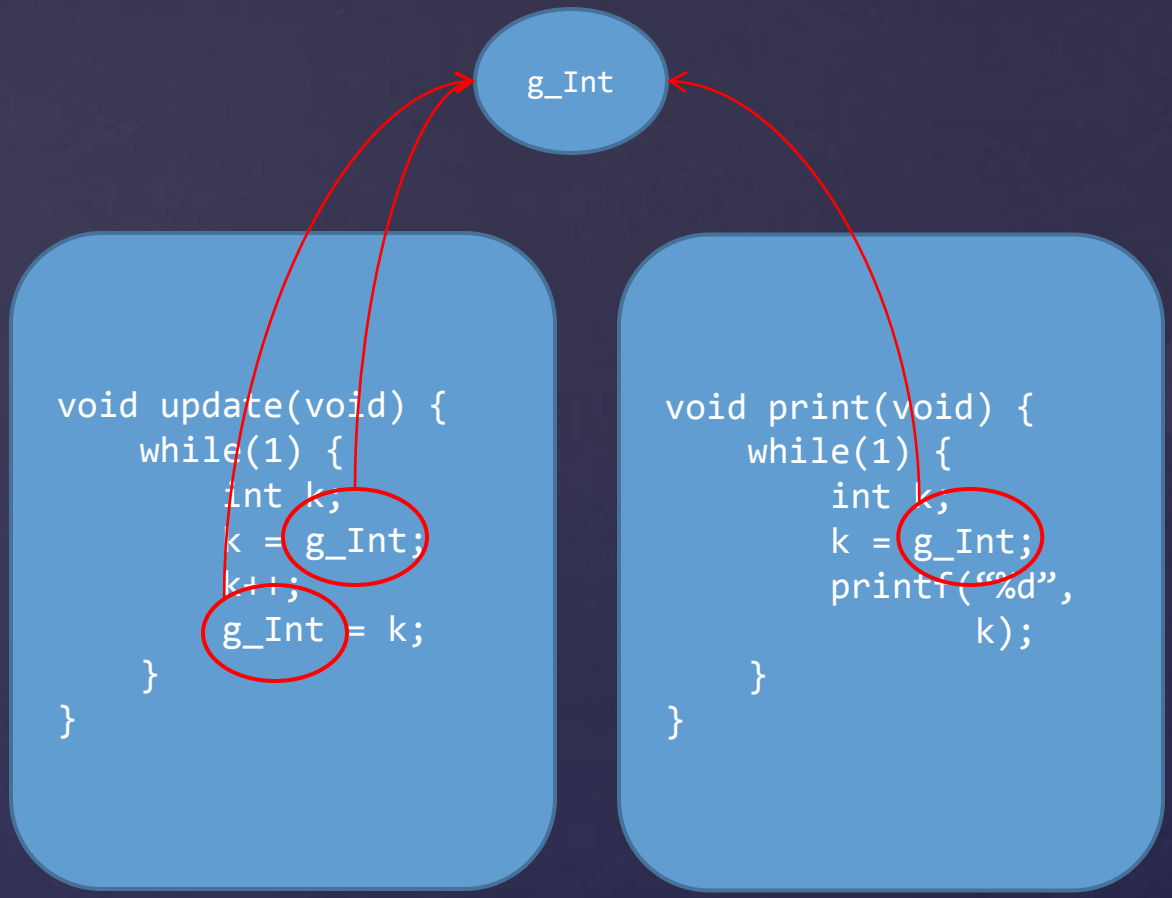

# CPU time is a resource

- Can you buy a computer with <2 CPUs now?
- The kernel is as multi-threaded as your applications are

# The kernel is highly multithreaded

# What is synchronization?

{




# Co-ordination

- Spinlock
- Semaphore
- Mutex

# Common synchronization tools

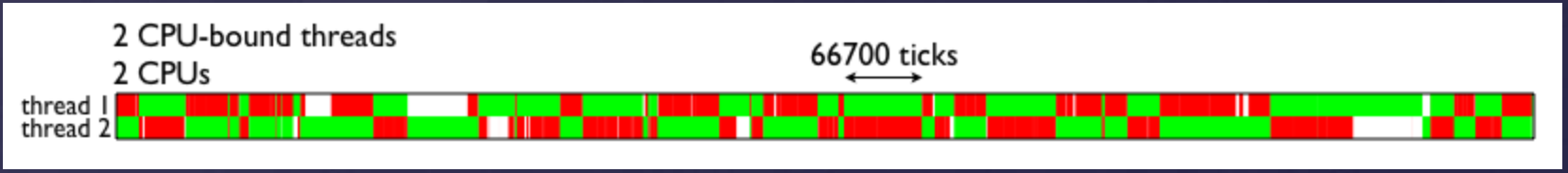
# Why is synchronization important?

{

# Prevention of chaos

- Concurrent data access is fraught with peril
- Many say “Don’t do it”
- Operating system developers have little choice

- Threads don't do anything useful while contending
- Time spent waiting for a lock is time that could be spent doing work




From: http://dabeaz.blogspot.com/2010/01/python-gil-visualized.html

# Overhead of synchronization

# What is RCU?

{ Read-Copy-Update

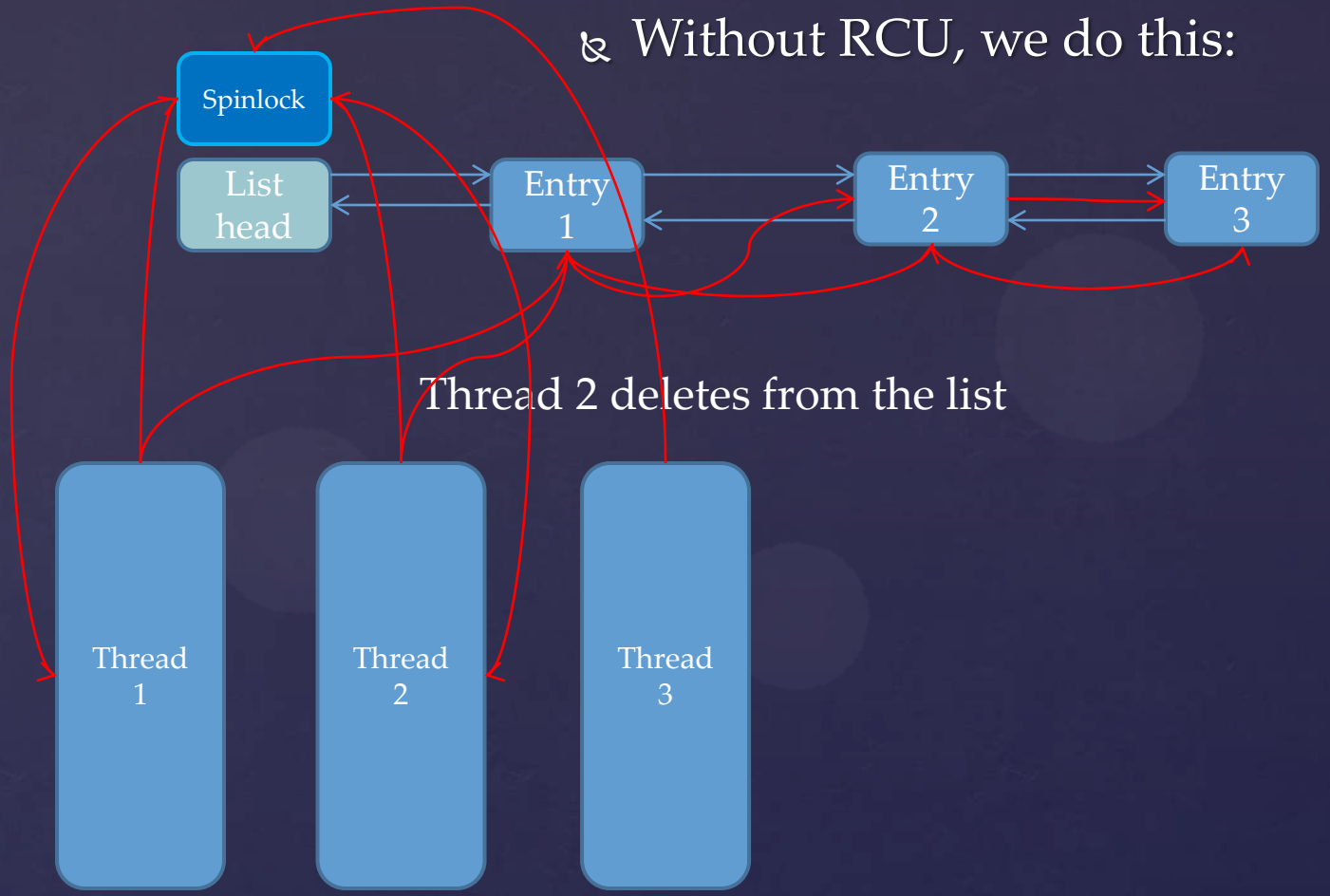
- A synchronization algorithm
- A type of mutual exclusion
- Readers access concurrently
- Protects readers from writers

# Wait-free synchronization

# How does RCU work?

{

- Without RCU, we do this:

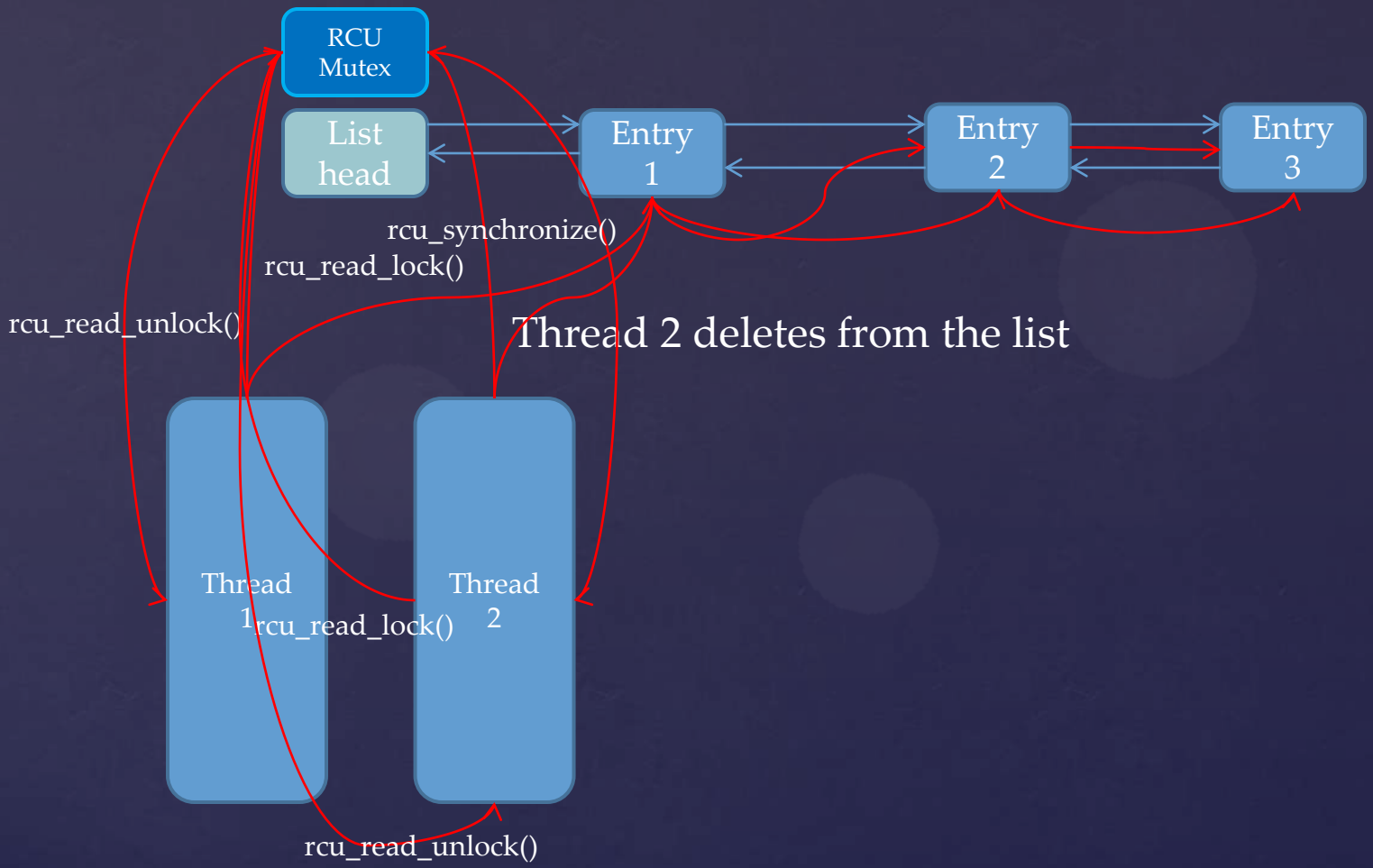



Thread 2 deletes from the list

Thread 3 reads from the list

Thread 1 reads from the list

- With RCU, we do this:




Thread 1 reads from the list

# Why is RCU important?

{

- Readers can “just go”
- Writers have to wait, but
- Writers are guaranteed safety
- Callback model allows writers to reclaim some speed

# Less time contending

# Still exploring its uses

- Applications of RCU are still being discovered
- Synchronization algorithms are an open field
- Many projects would benefit