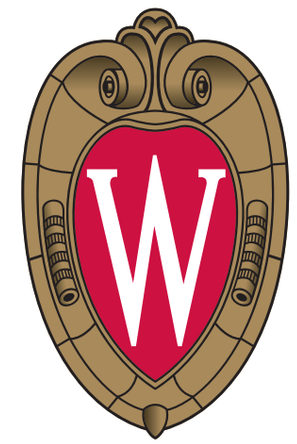
Virtual machine-provided
Context sensitive page mappings

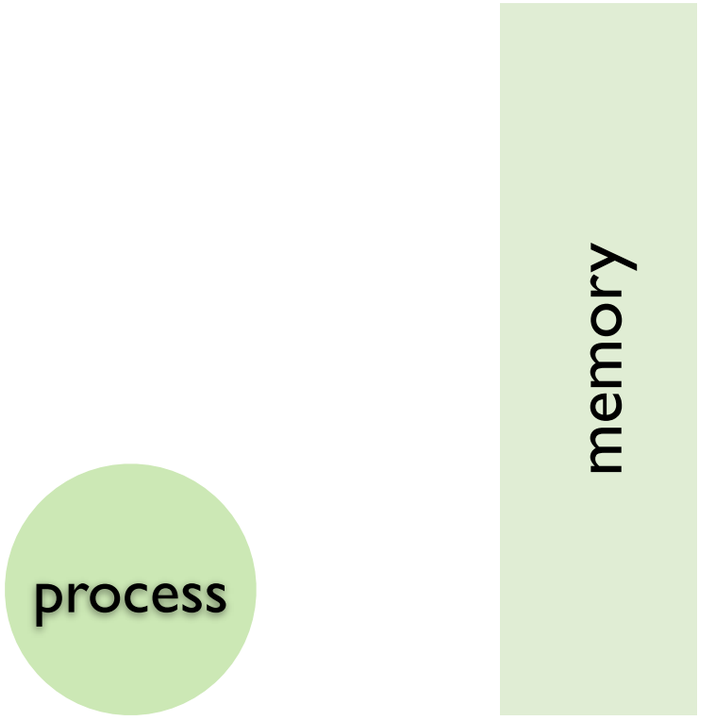
Nathan Rosenblum, Gregory Cooksey, Barton P. Miller
Computer Sciences Department
University of Wisconsin

{nater,cooksey,bart}@cs.wisc.edu
http://pages.cs.wisc.edu/~nater/



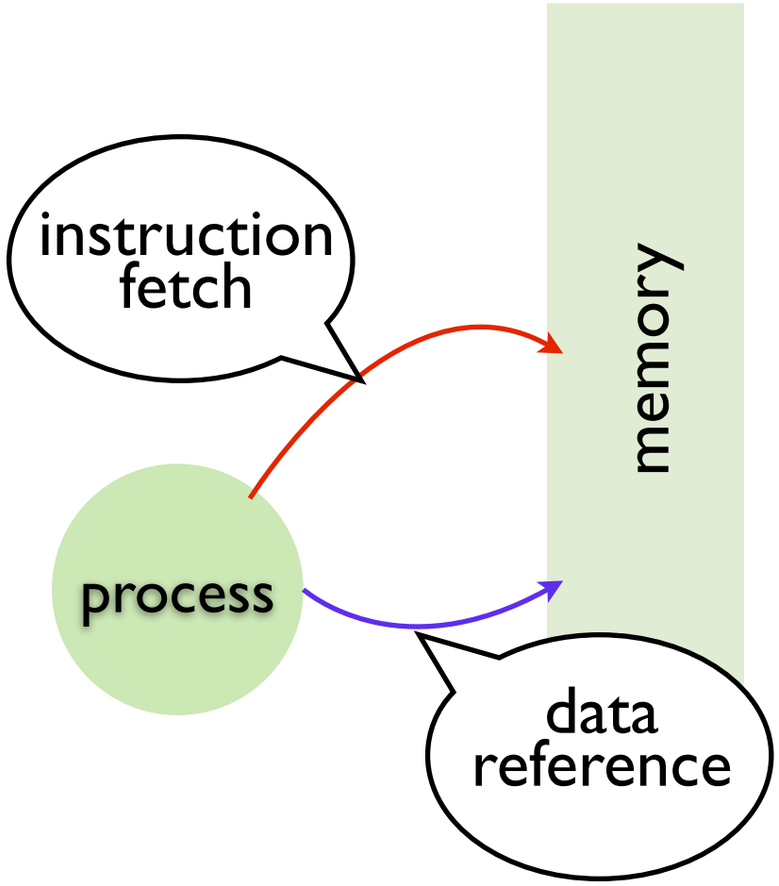
THE UNIVERSITY of WISCONSIN MADISON

Context



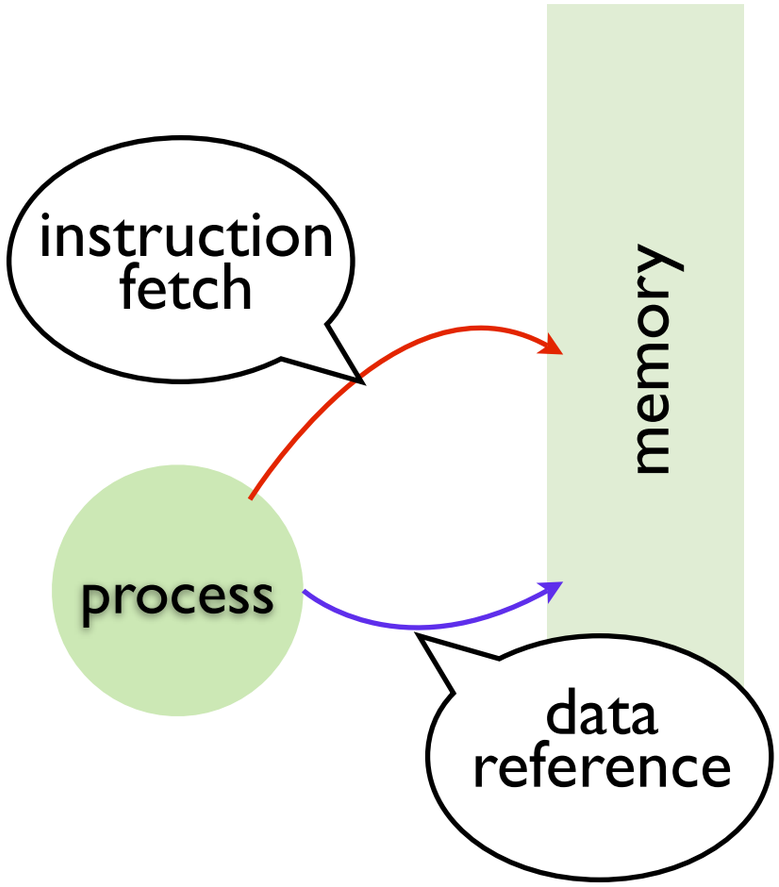
single store (von Neumann)
memory architecture

Context



single store (von Neumann) memory architecture

Context



single store (von Neumann) memory architecture

introspective (self-examining) code

hot/runtime patched code

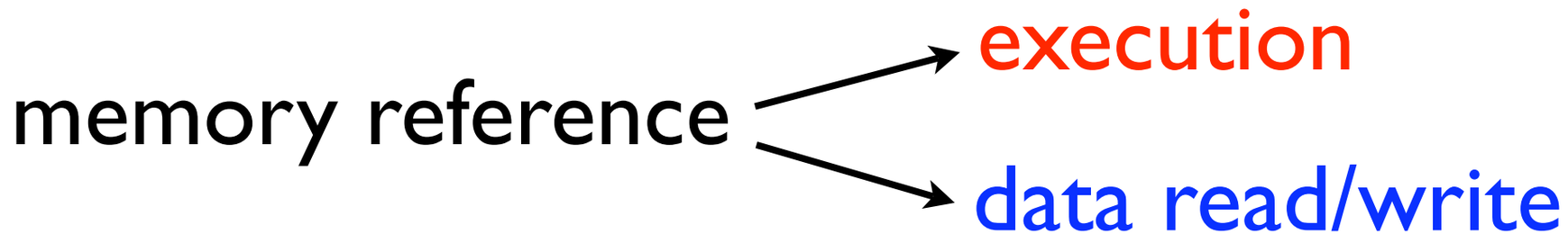
self-modifying programs

dynamic code unpacking/ decryption

What is context sensitivity?

What ~~is~~ context sensitivity?

kind of



Context sensitivity has been useful

Circumvention

Wurster, van Oorschot, Somayaji. *A generic attack on checksumming-based software tamper resistance*. IEEE Symposium on Security and Privacy. 2005

Stealth

Sparks, Butler. *Shadow Walker: Raising the bar for rootkit detection*. Black Hat Japan. 2005

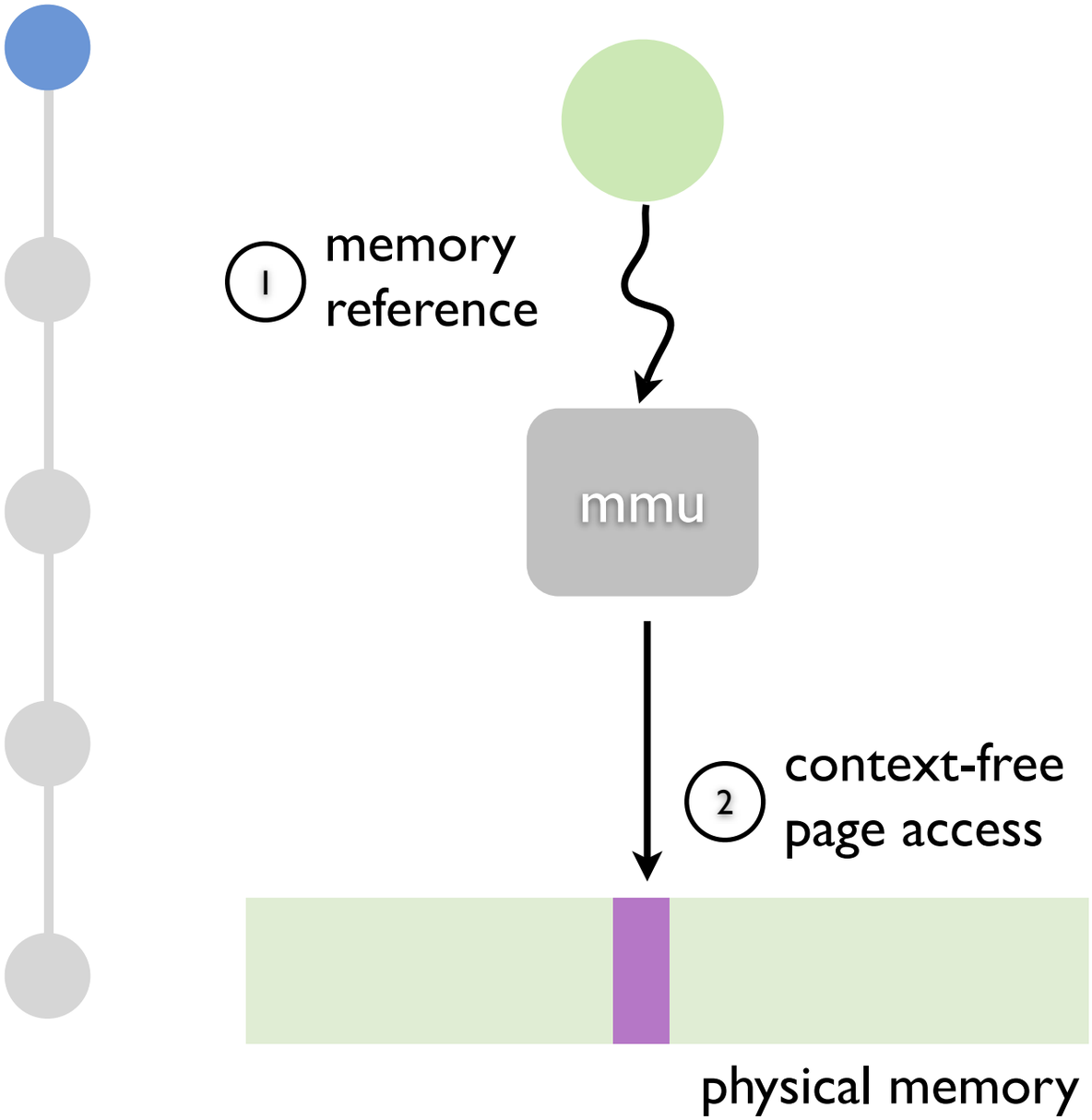
Protection

PAX Team. *PaX*. http://pax.grsecurity.net

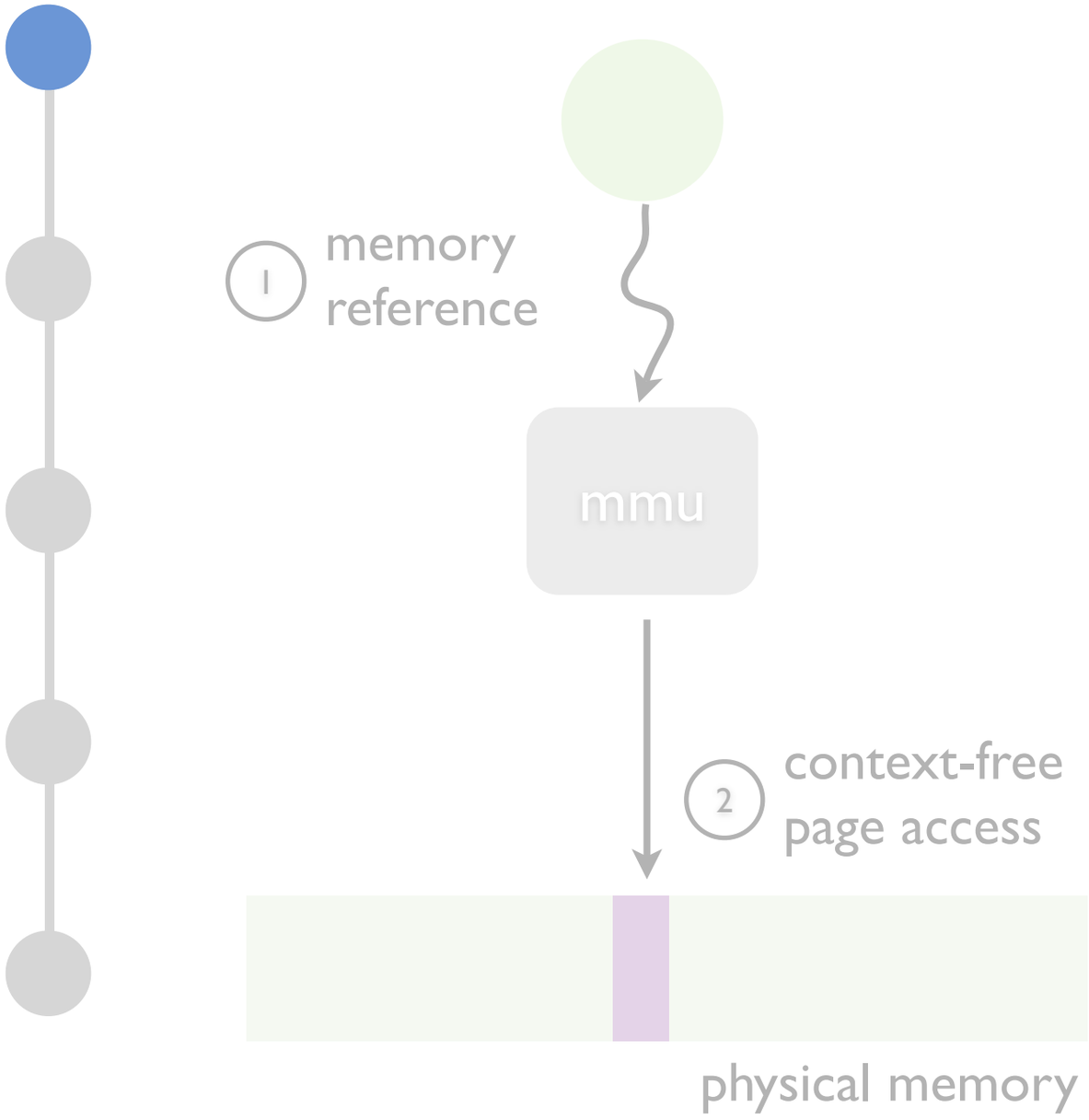
A brief look ahead

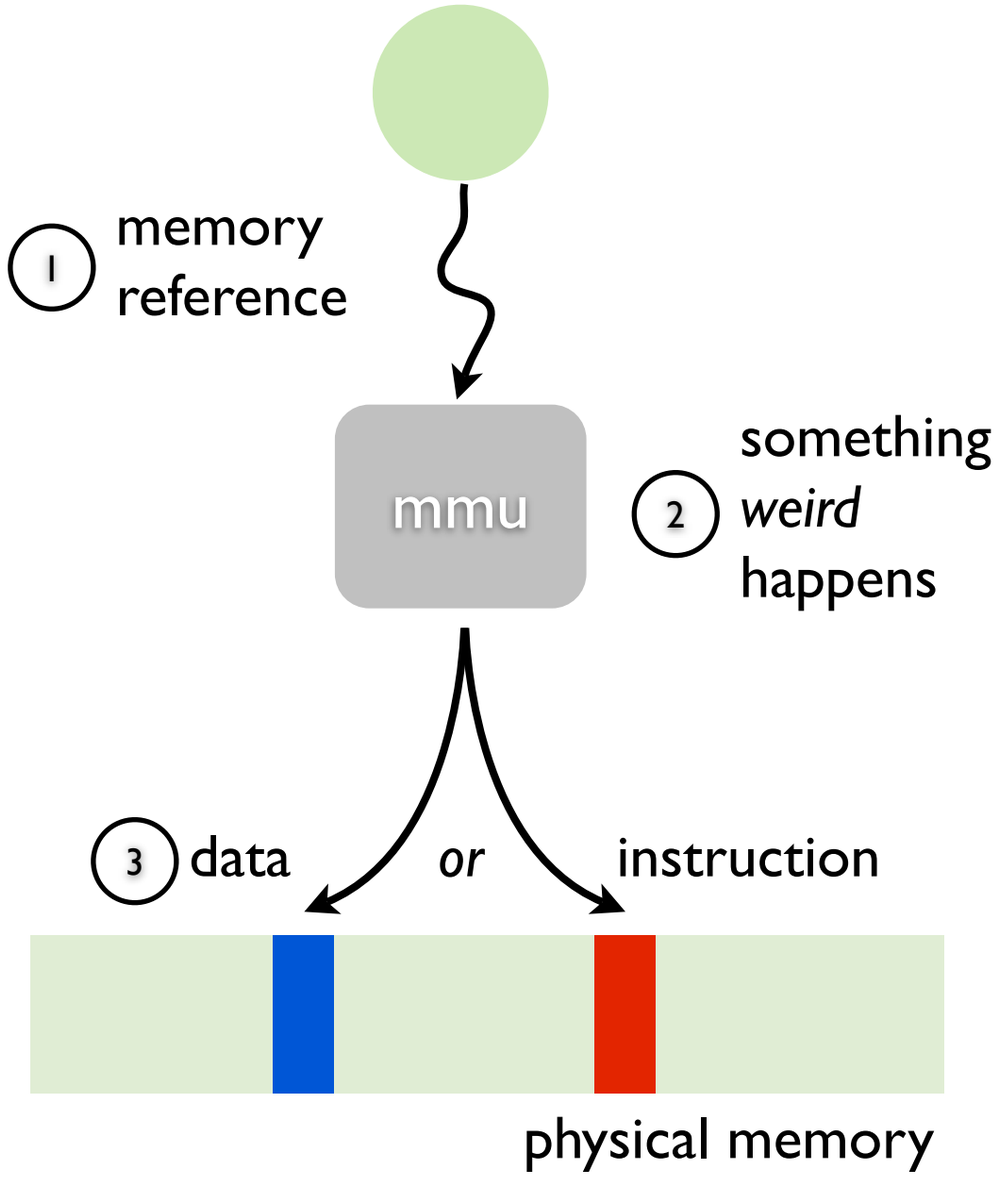
- What are context sensitive page mappings?
- Why "virtual machine [monitor]-provided"?
- Nitty-gritty details
- Case study: self-checksumming codes
- What else are CSPM good for?

Context sensitive page mappings

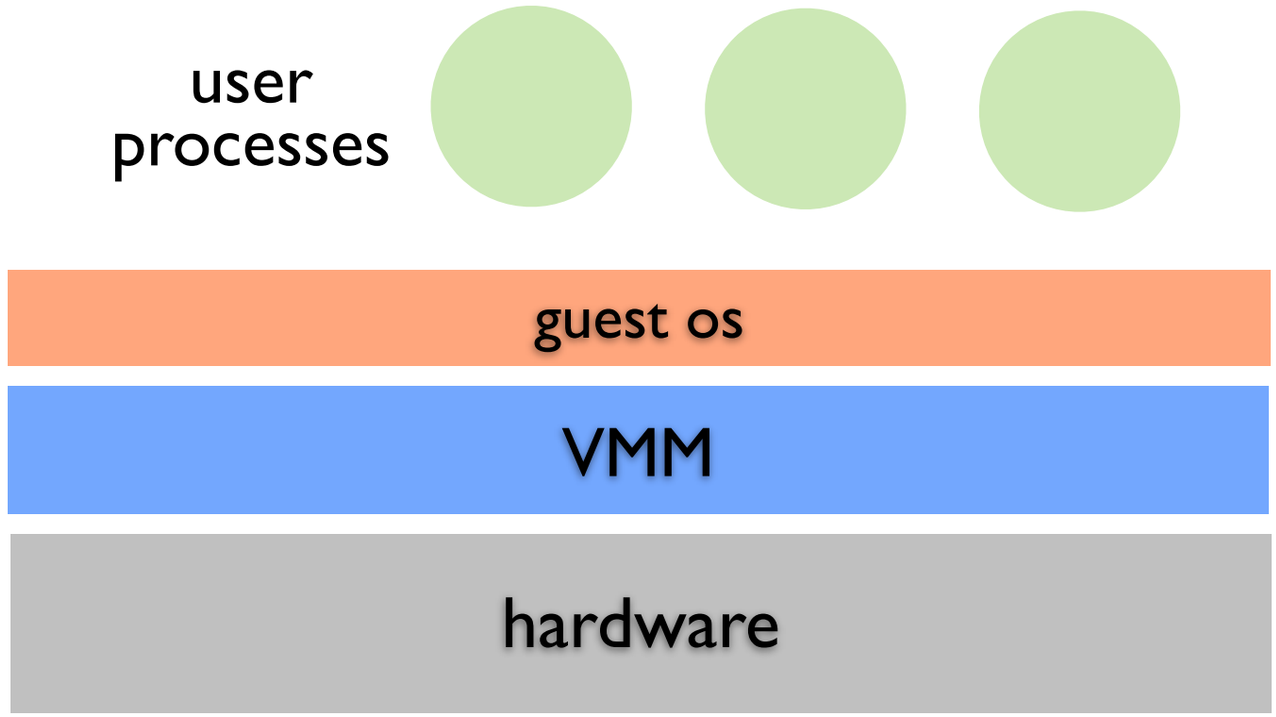


Context sensitive page mappings

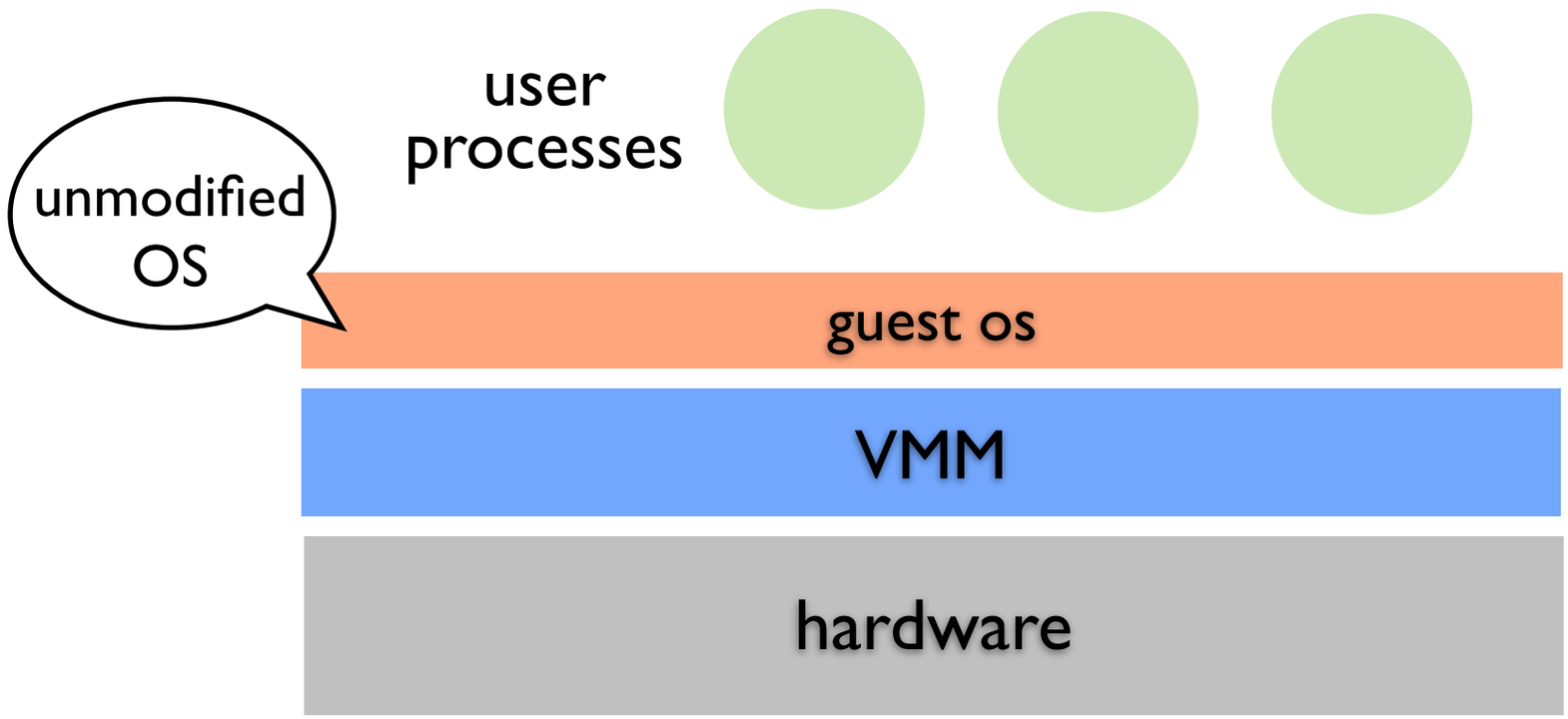




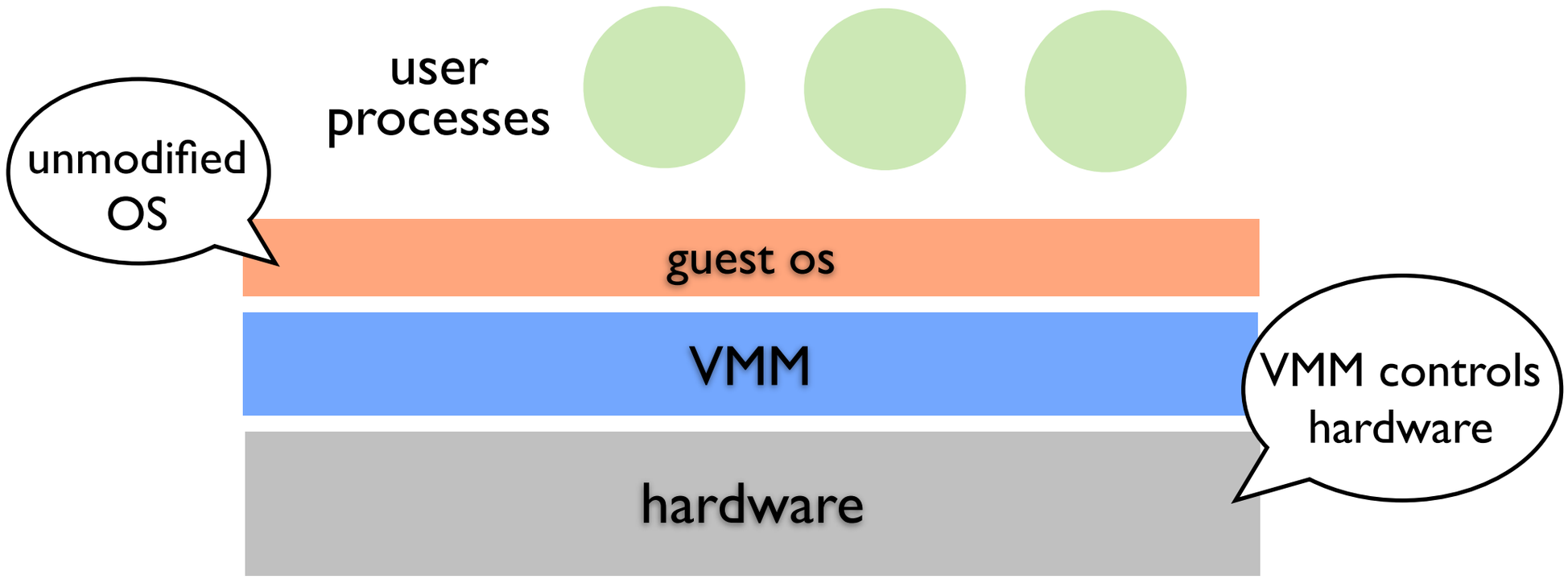
Why implement with VMMs?



Why implement with VMMs?

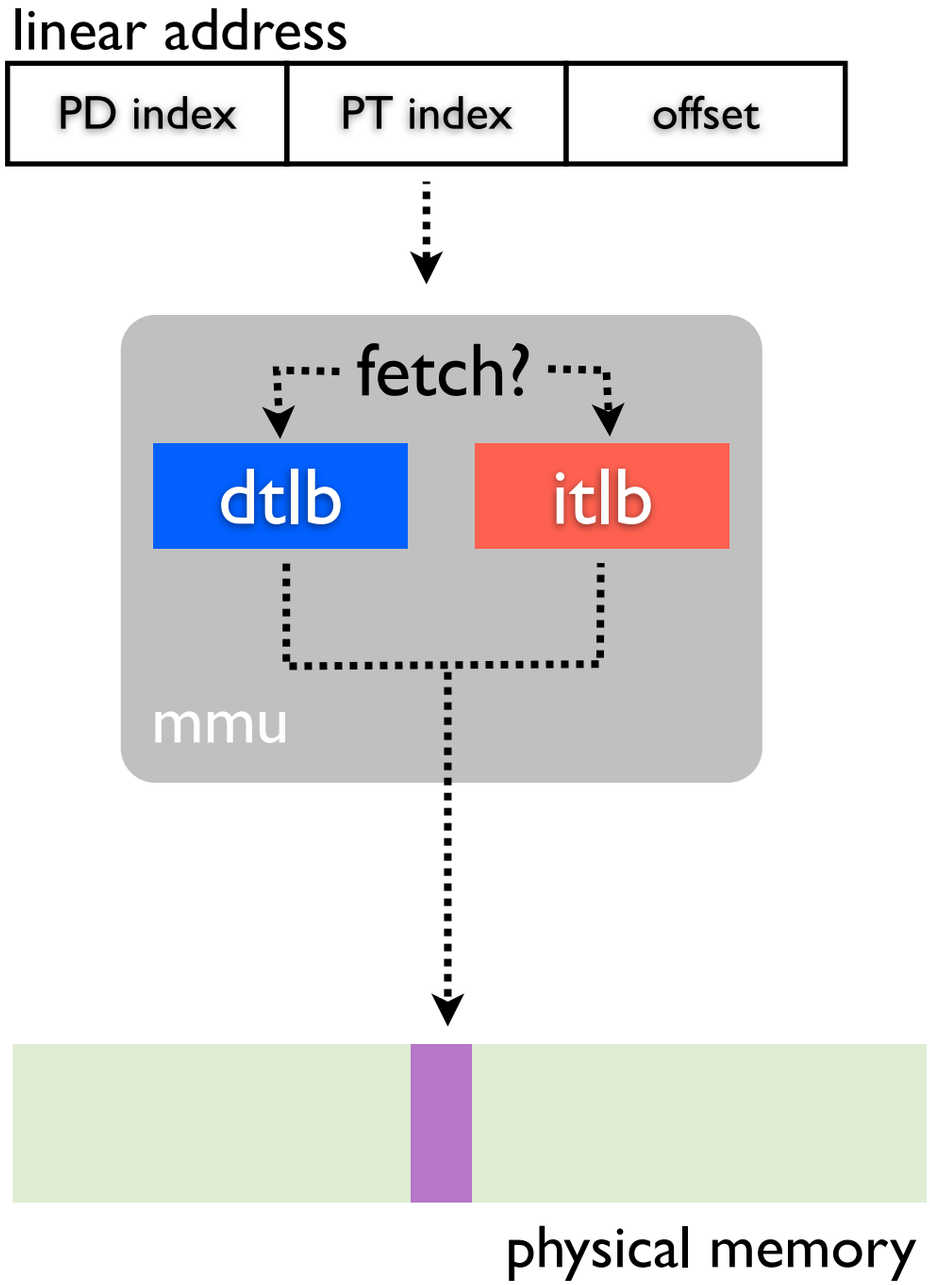


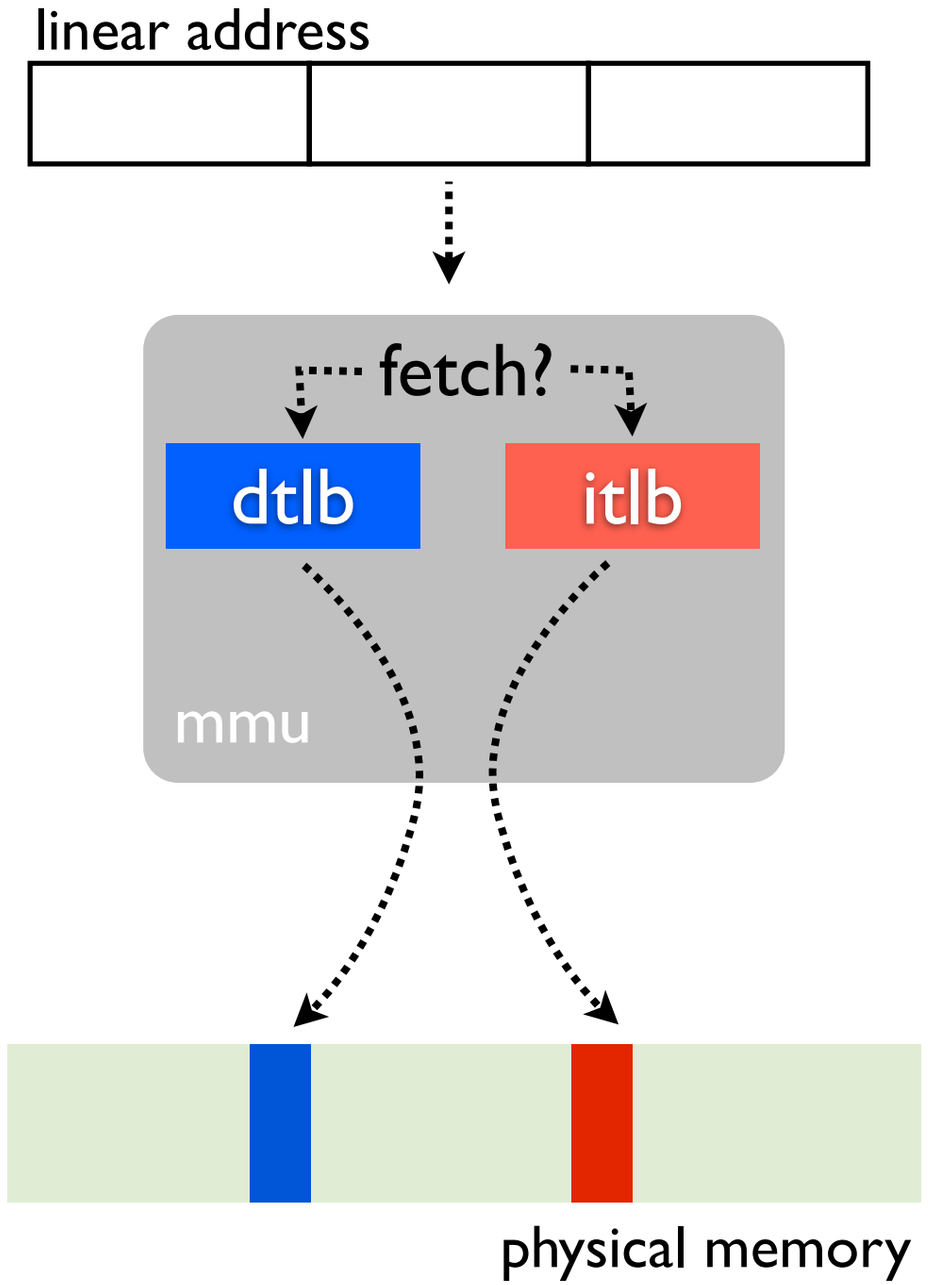
Why implement with VMMs?



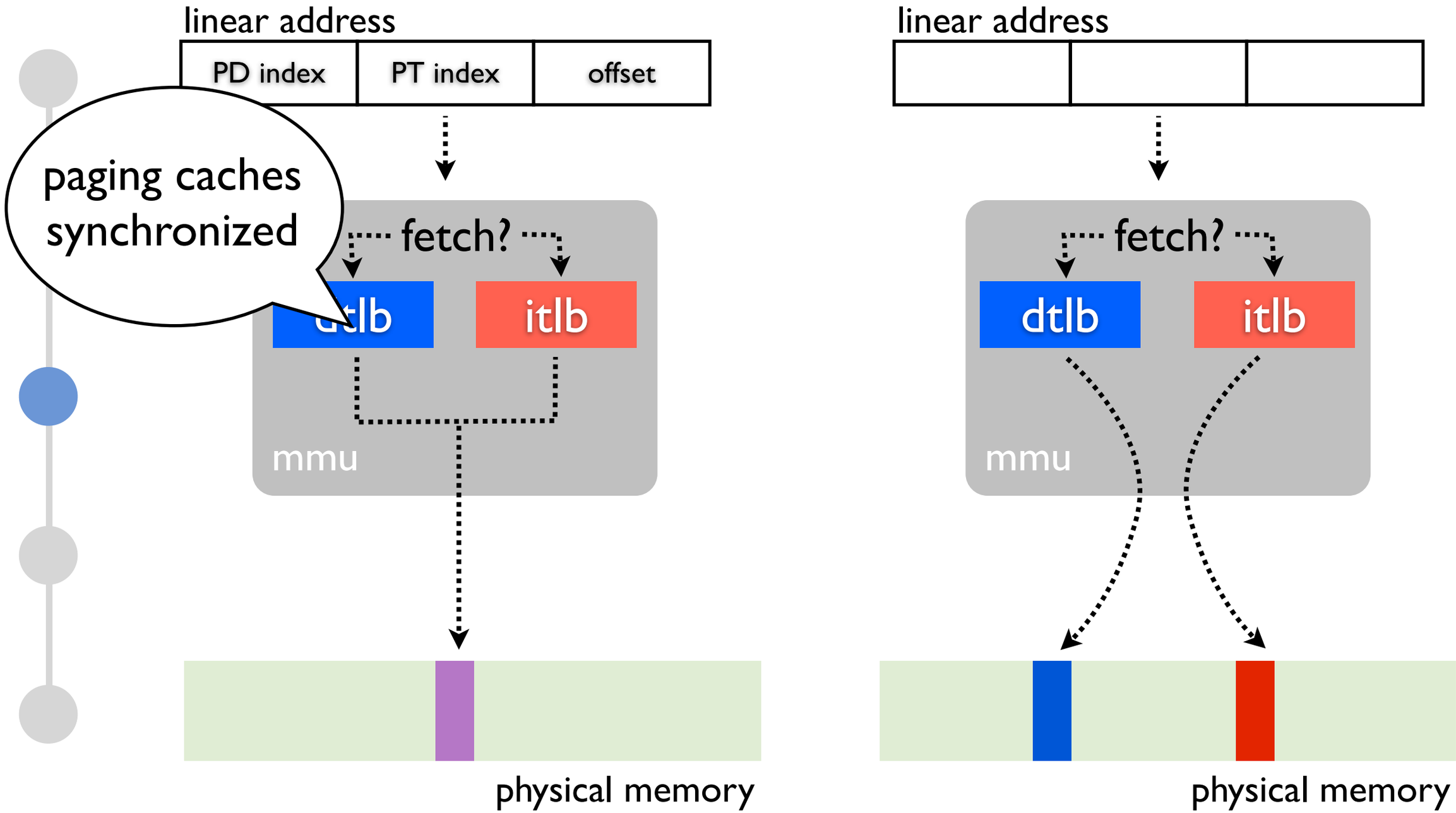
protection → control of memory management hardware

Splitting paged memory on IA32

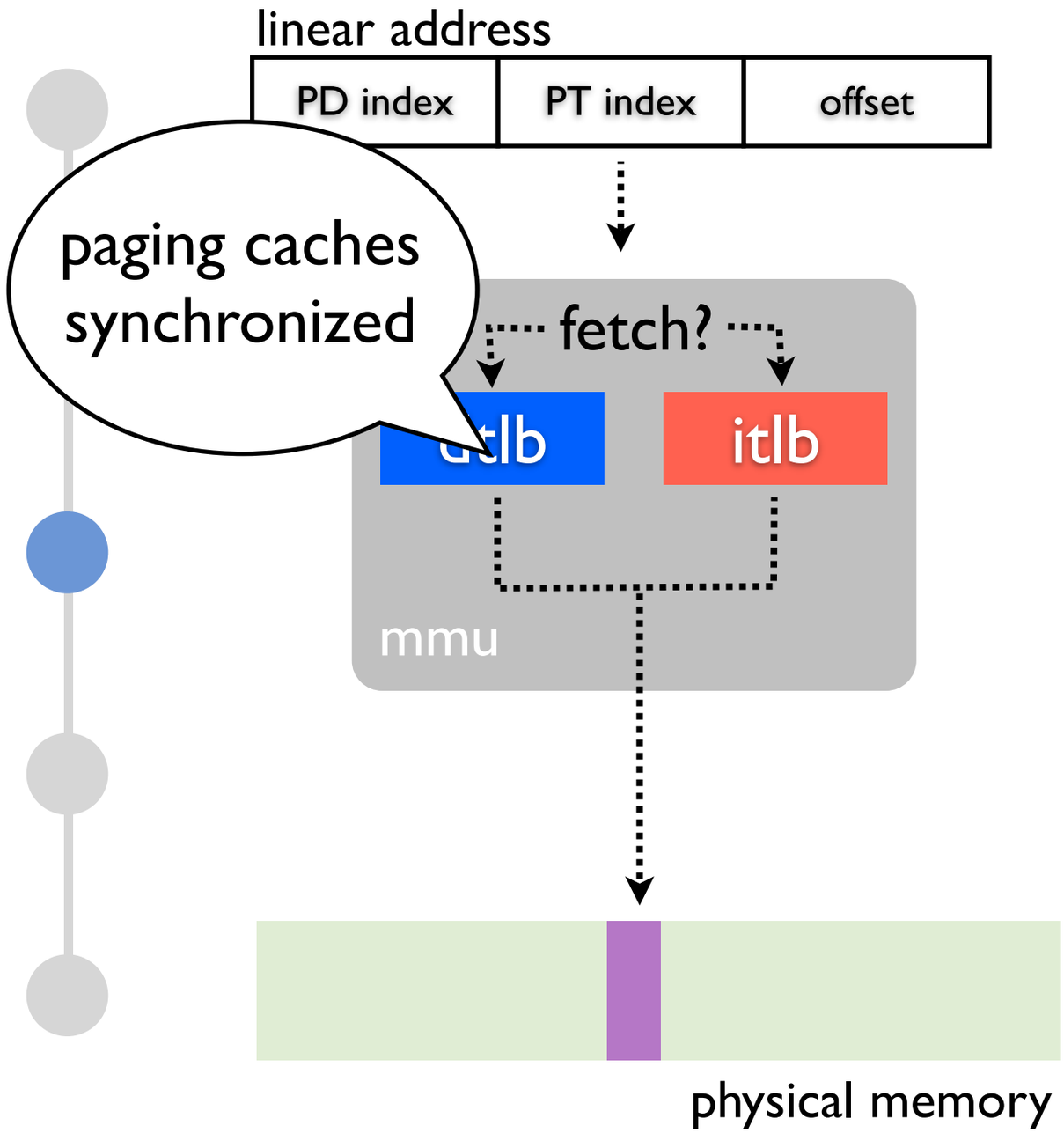


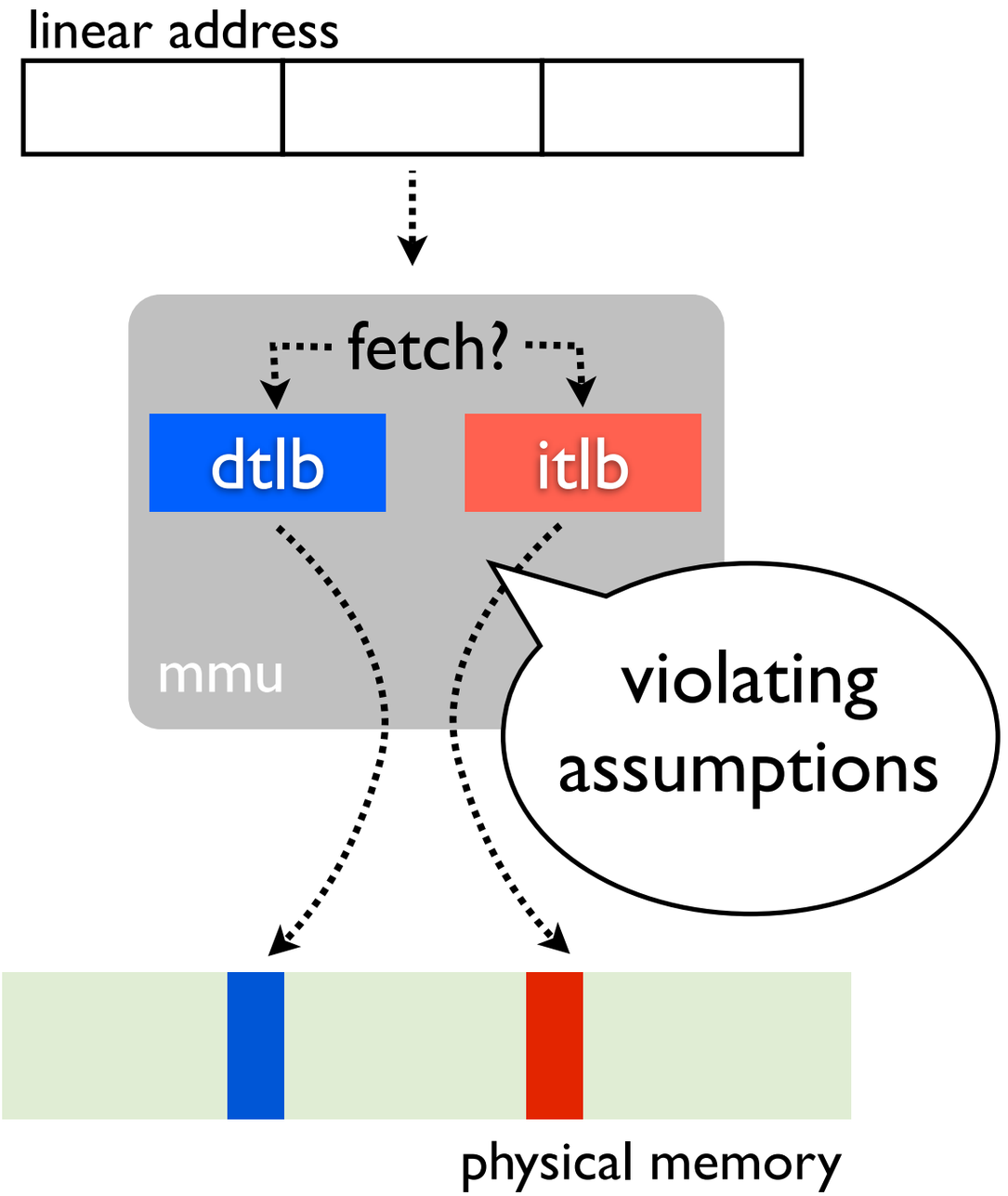


Splitting paged memory on IA32



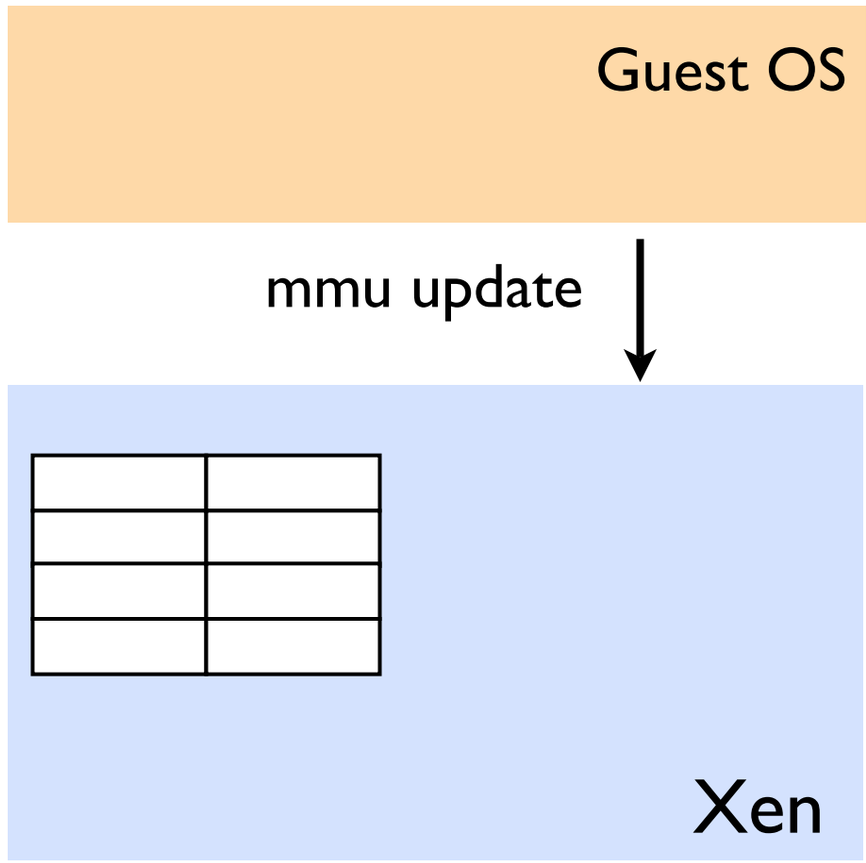
Splitting paged memory on IA32





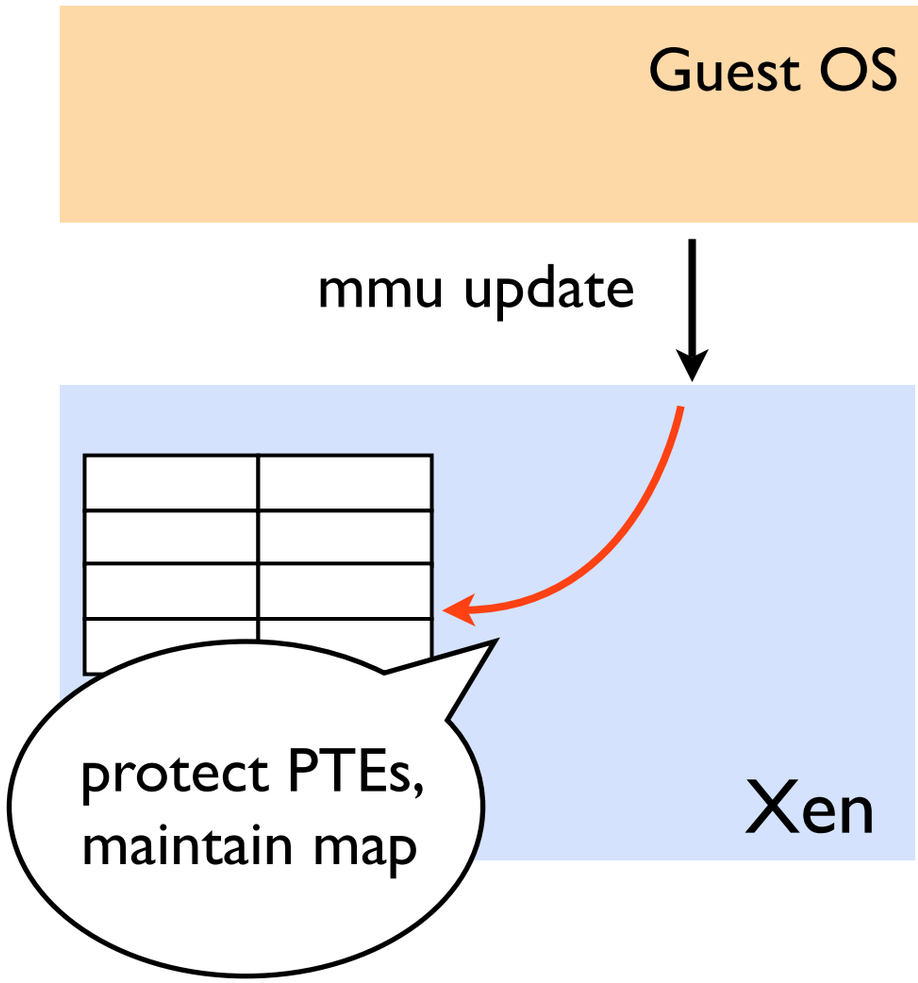
Xen hypervisor implementation

MMU Management



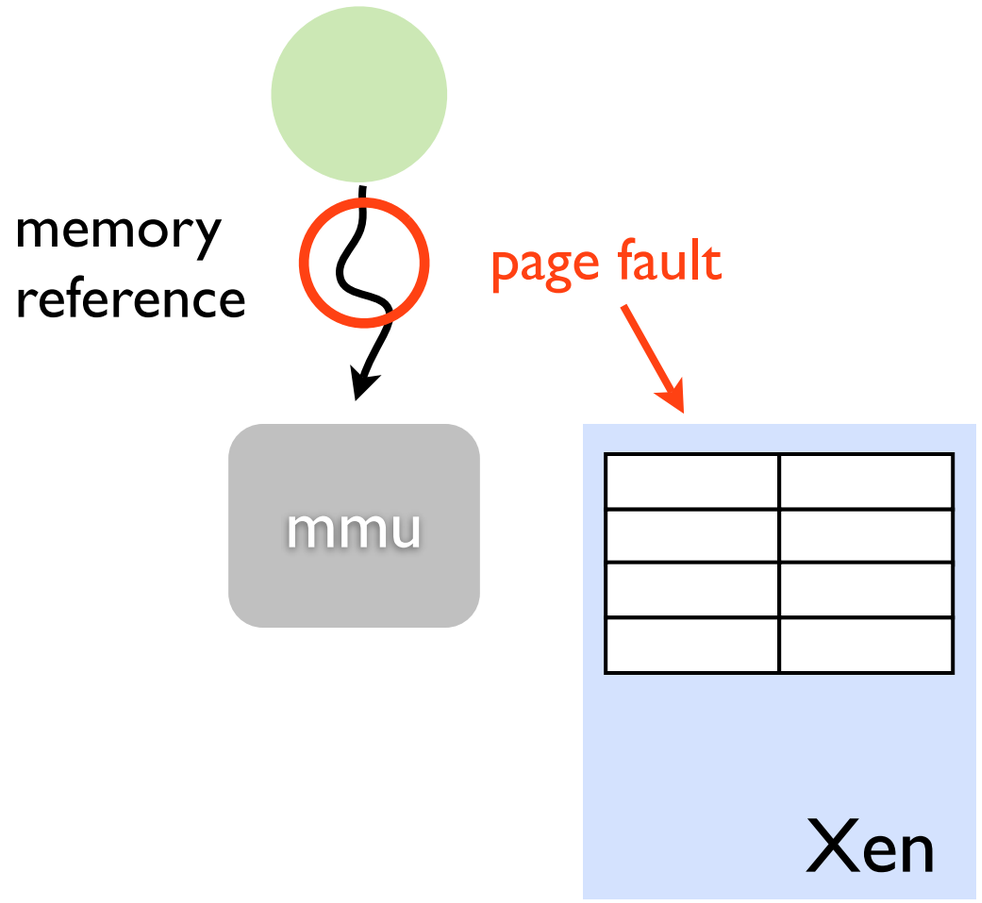
Xen hypervisor implementation

MMU Management



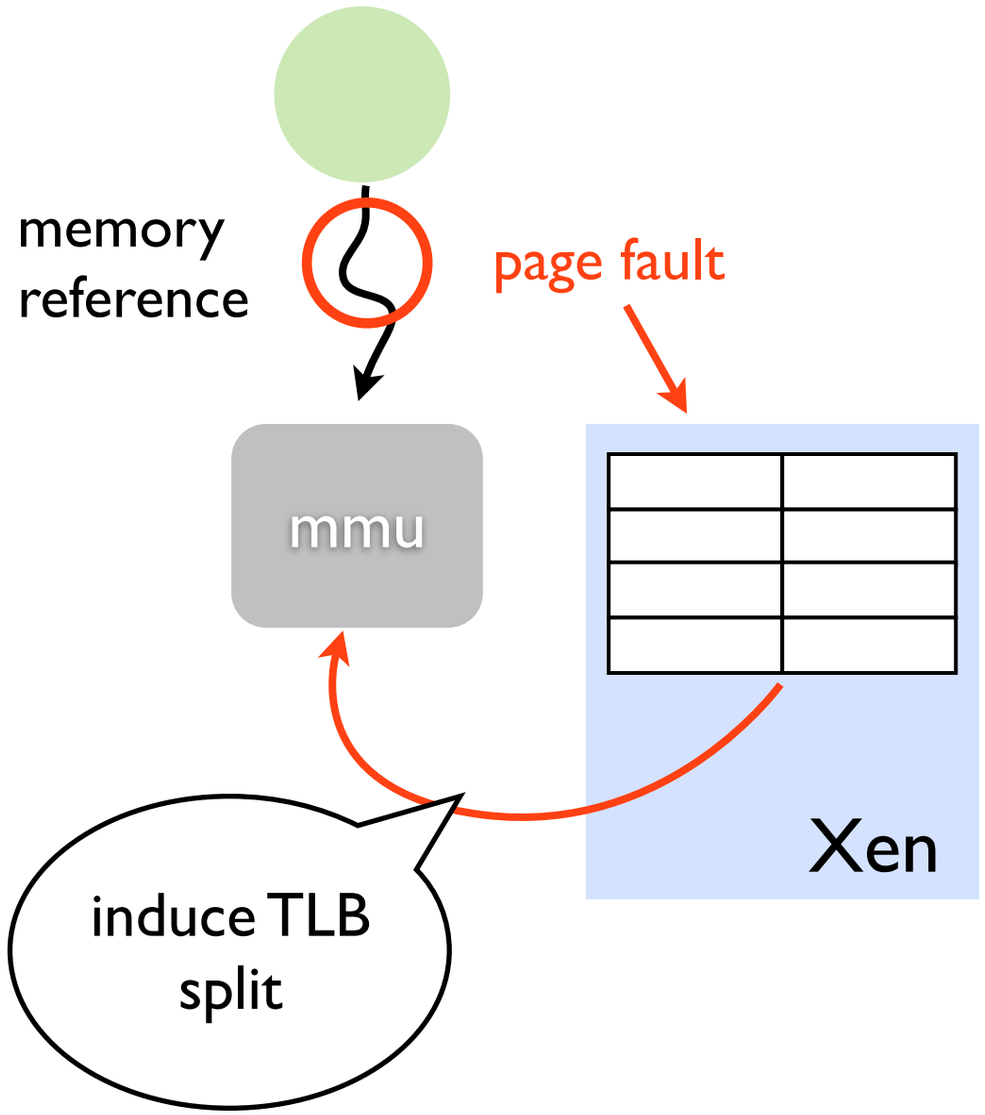
Xen hypervisor implementation

Page Fault Handling

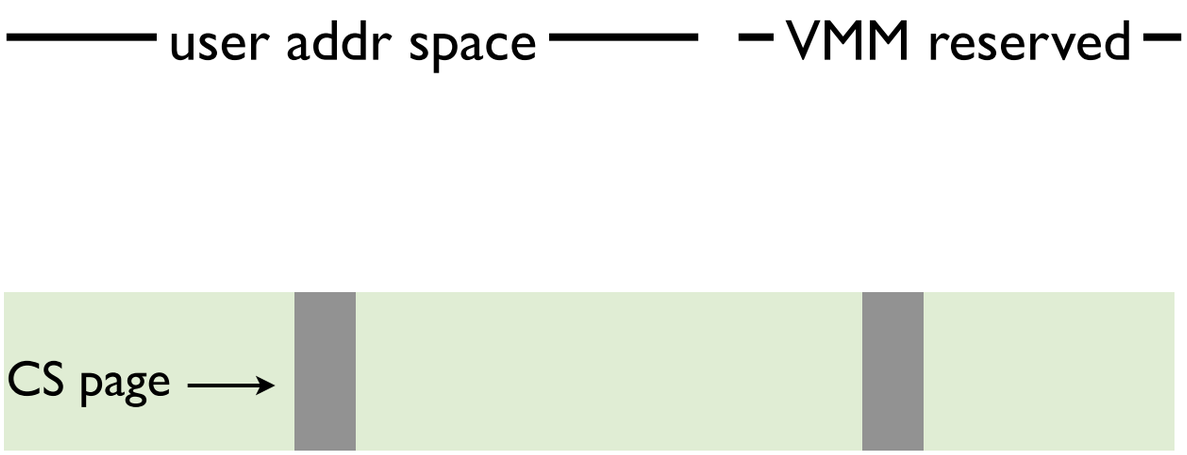


Xen hypervisor implementation

Page Fault Handling

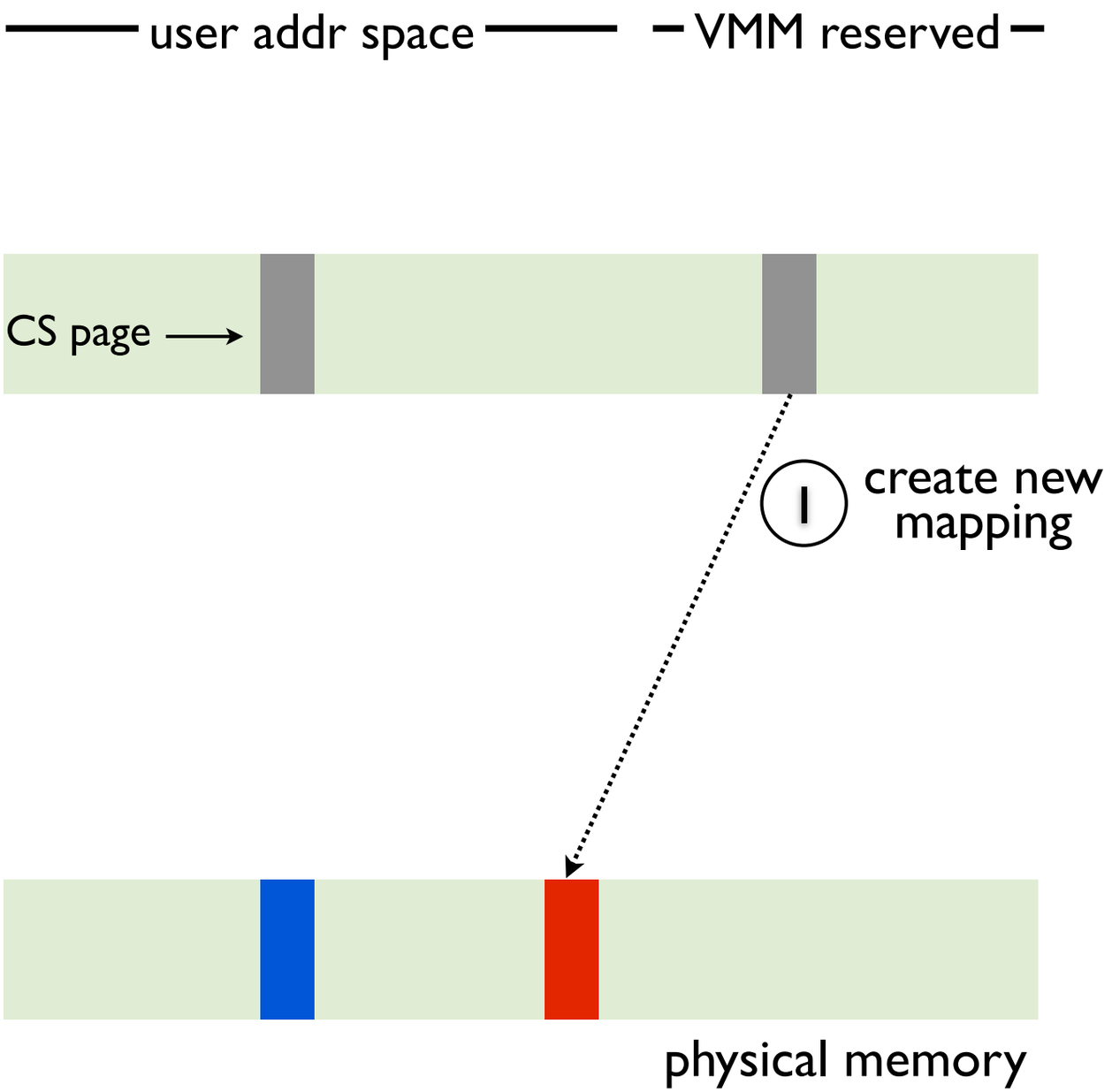


Desynchronizing TLBs



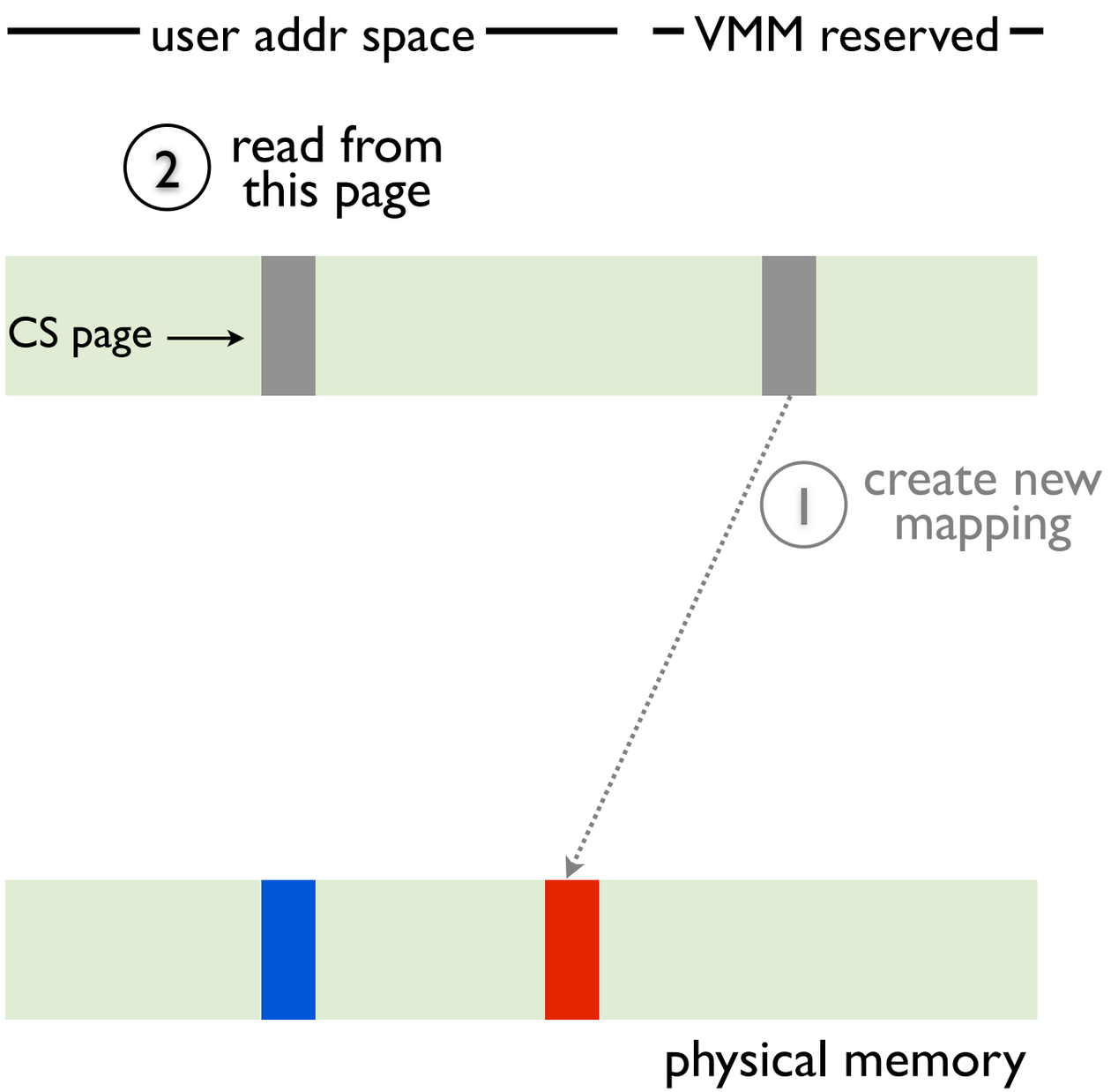
Flush TLB on page fault, then:

Desynchronizing TLBs



Flush TLB on page fault, then:

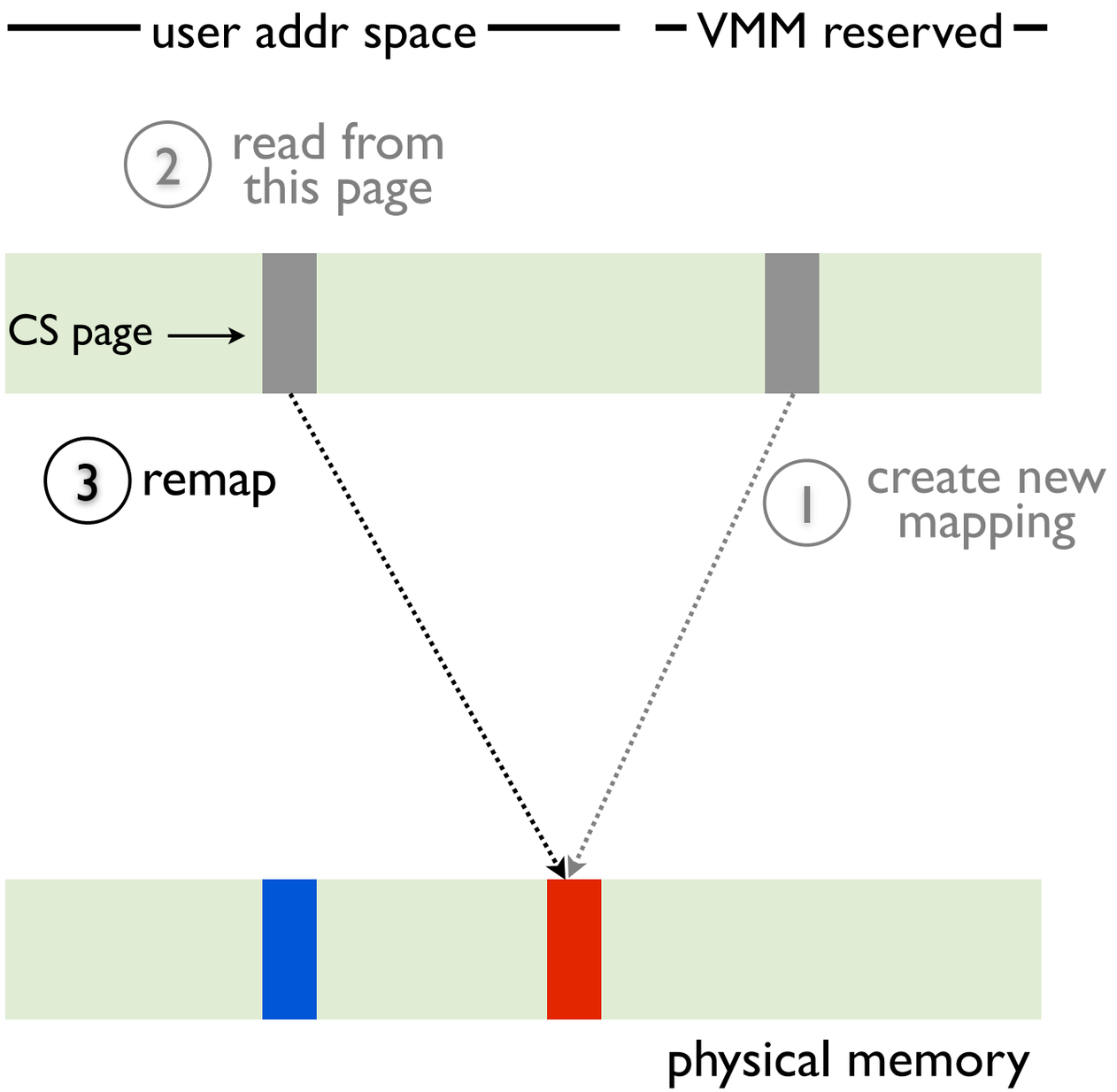
Desynchronizing TLBs



Flush TLB on page fault, then:

DTLB loaded

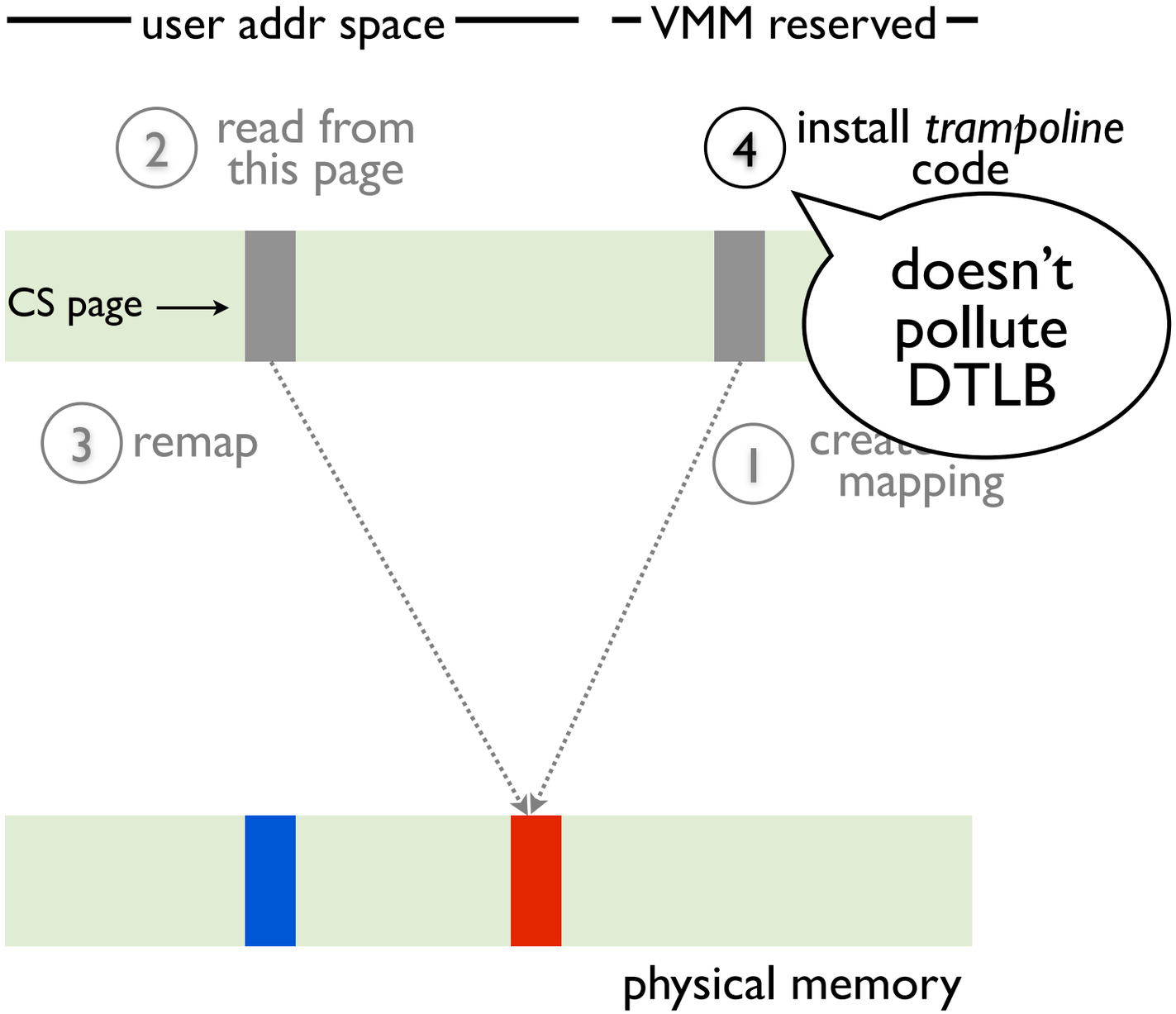
Desynchronizing TLBs



Flush TLB on page fault, then:

DTLB loaded

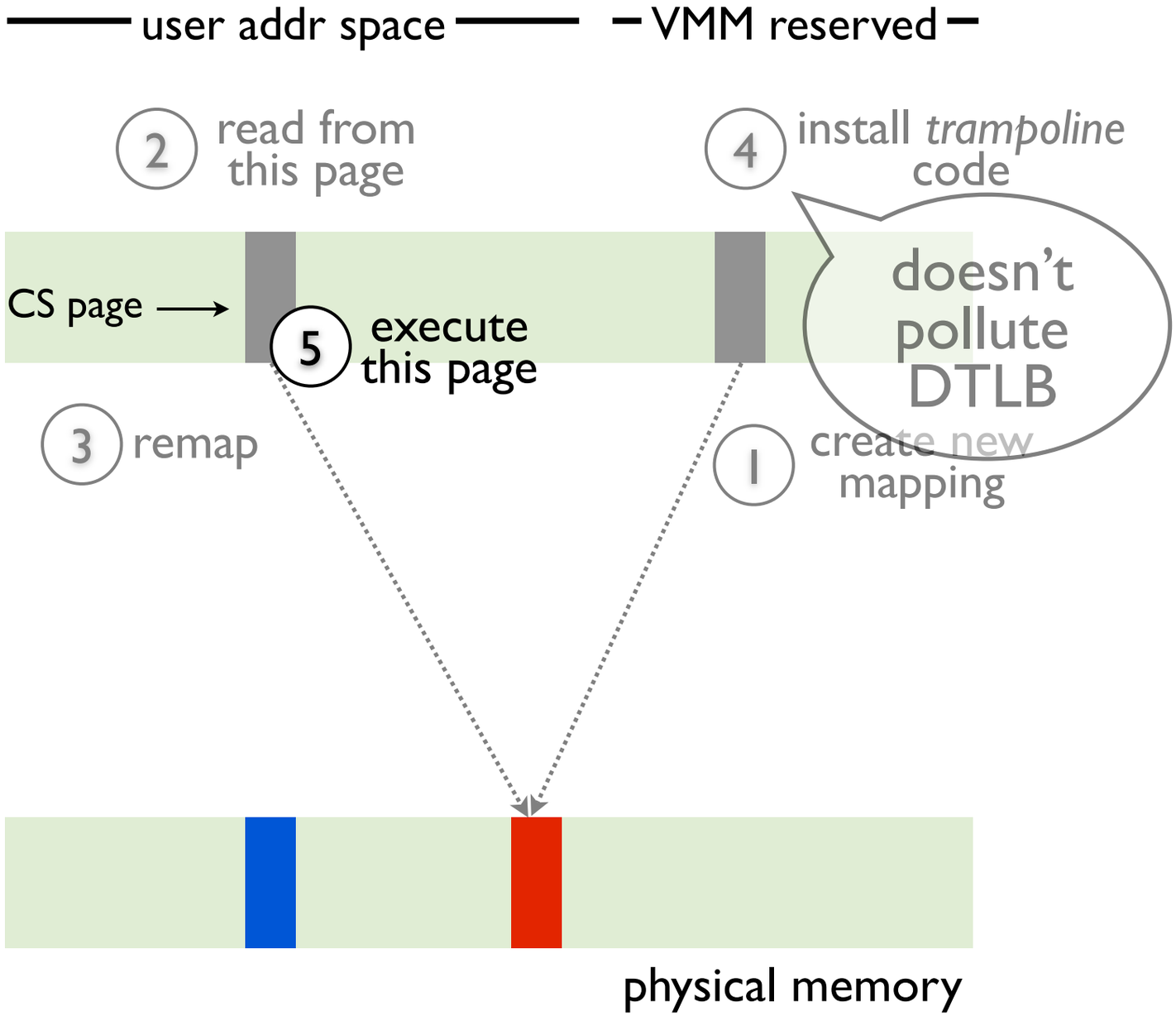
Desynchronizing TLBs



Flush TLB on page fault, then:

DTLB loaded

Desynchronizing TLBs

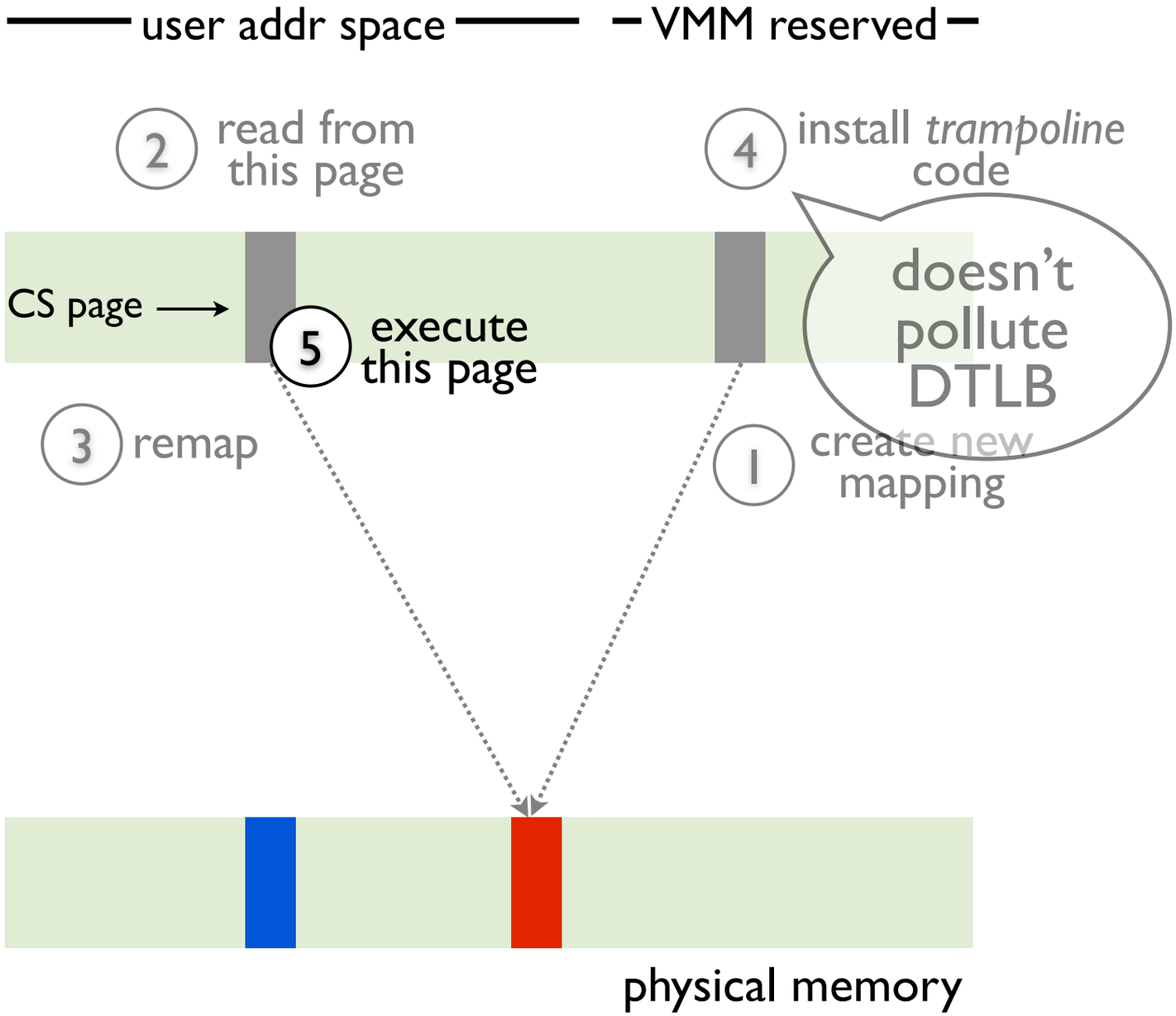


Flush TLB on page fault, then:

DTLB loaded

ITLB loaded

Desynchronizing TLBs



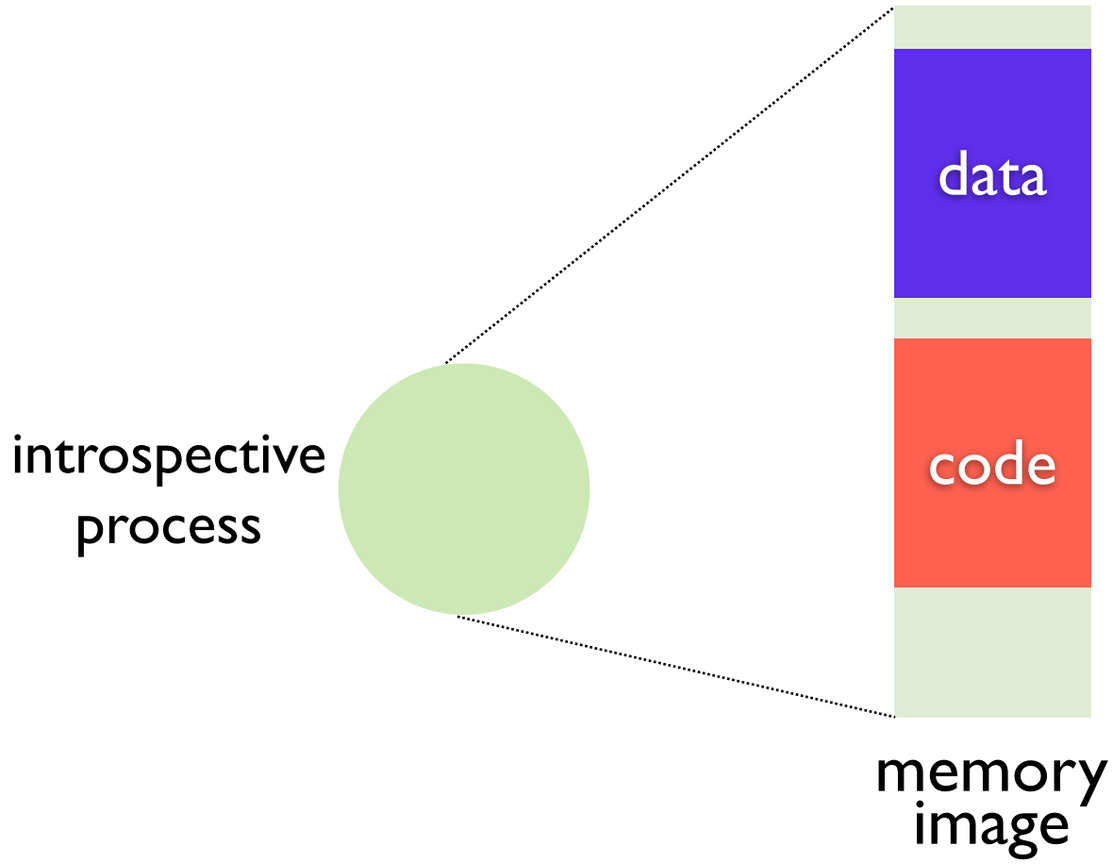
Flush TLB on page fault, then:

DTLB loaded

ITLB loaded

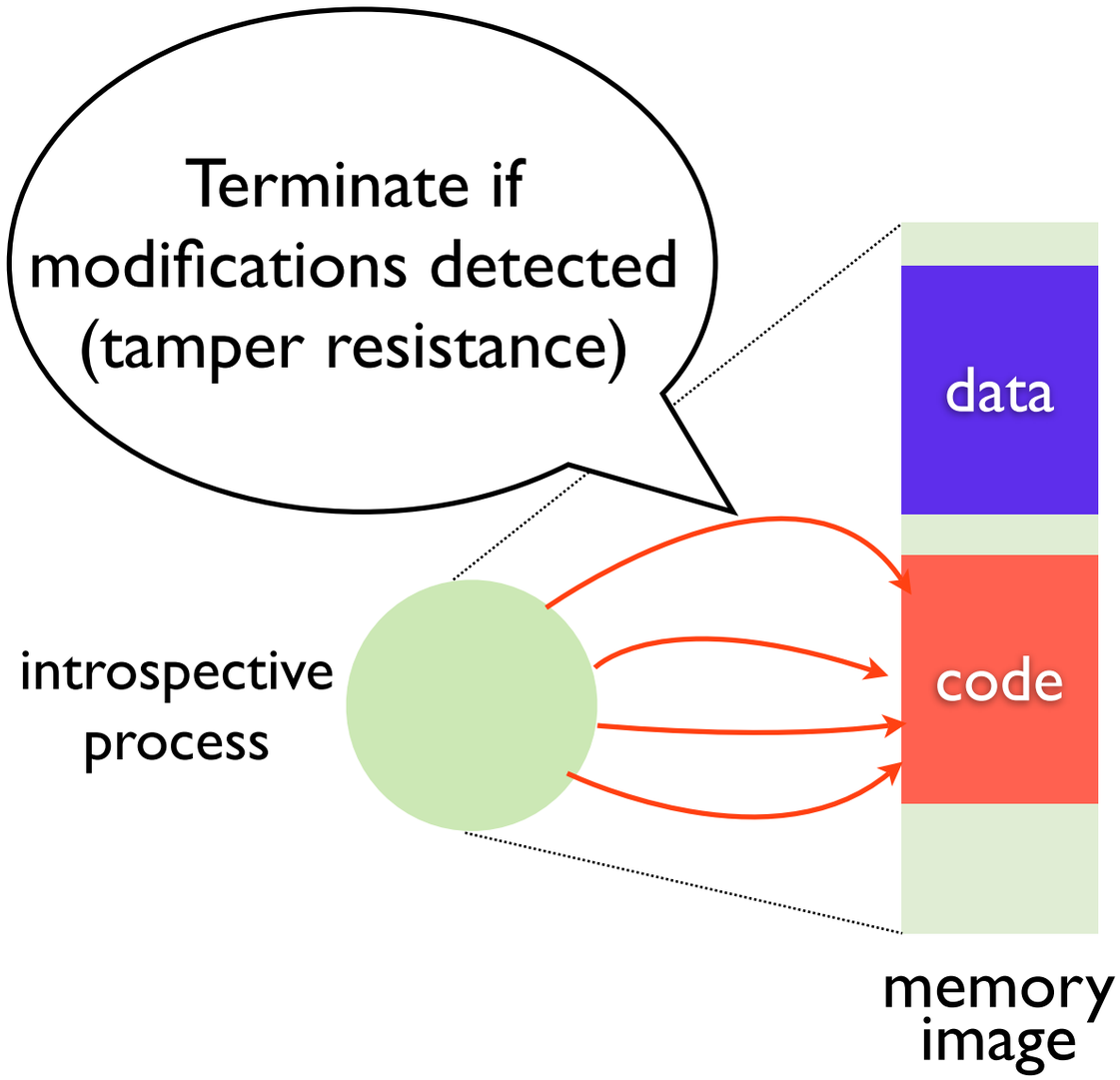
Restore mappings and return

Self checksumming code



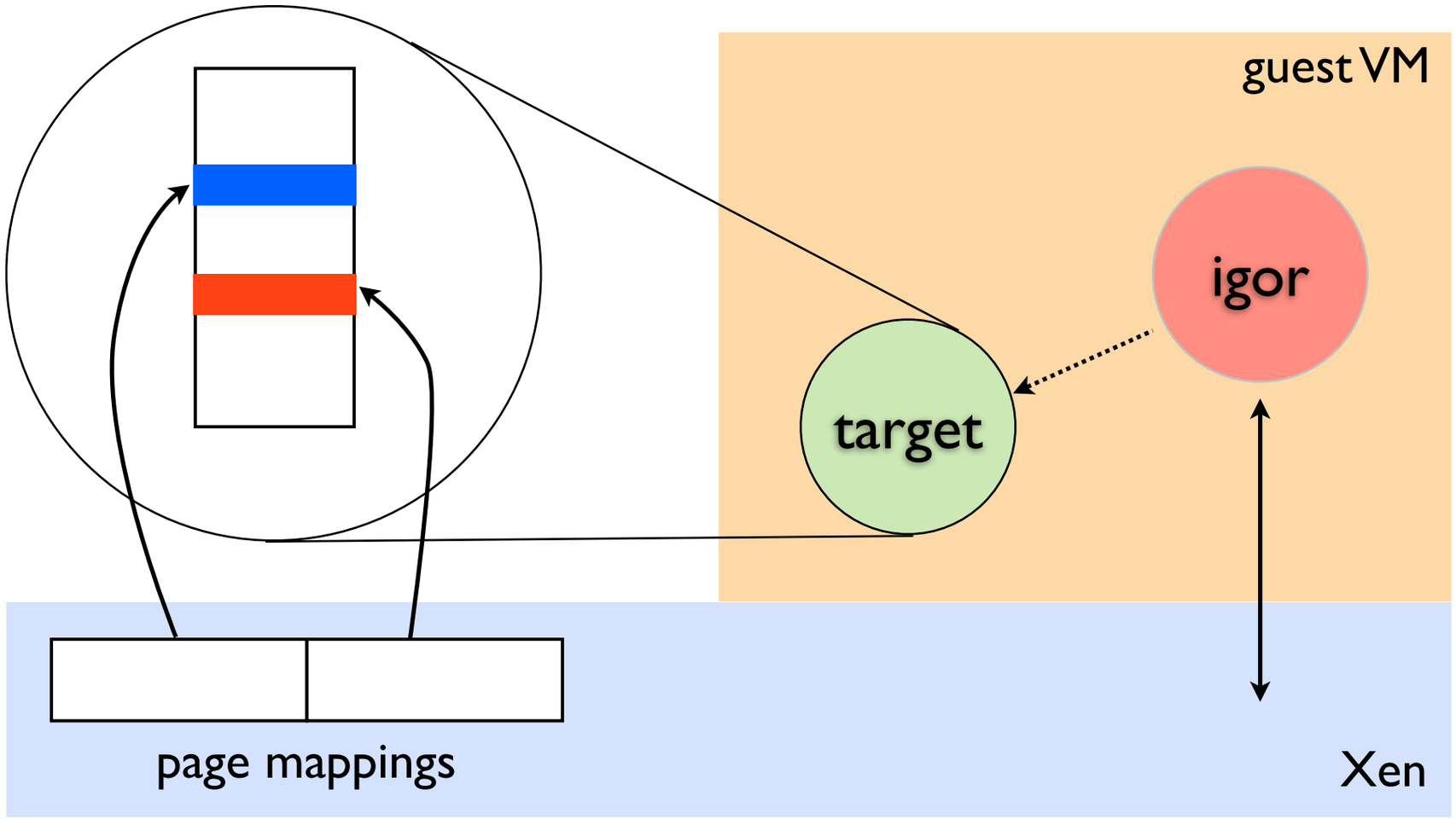
Assumes that bytes read and bytes executed are the same

Self checksumming code

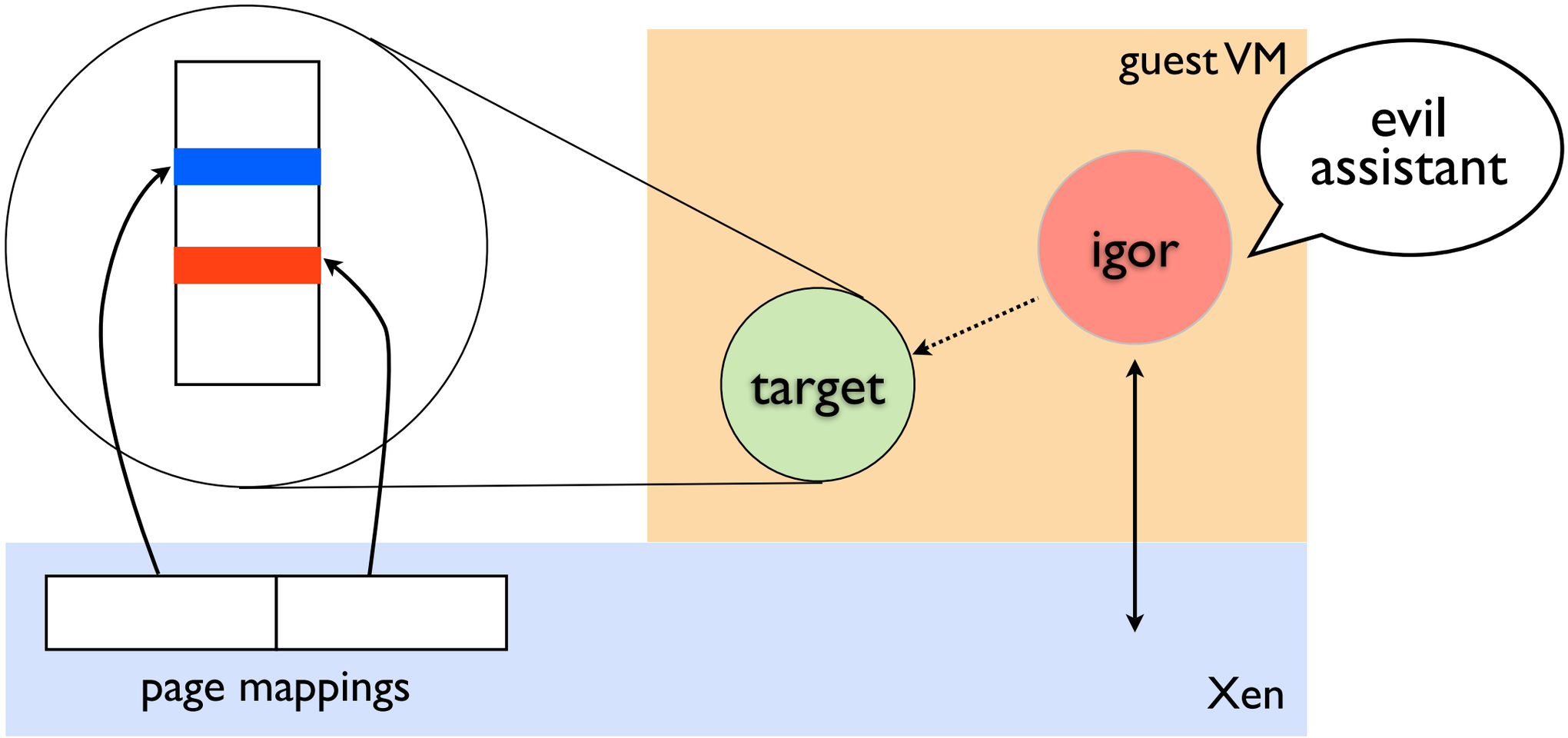


Assumes that bytes read and bytes executed are the same

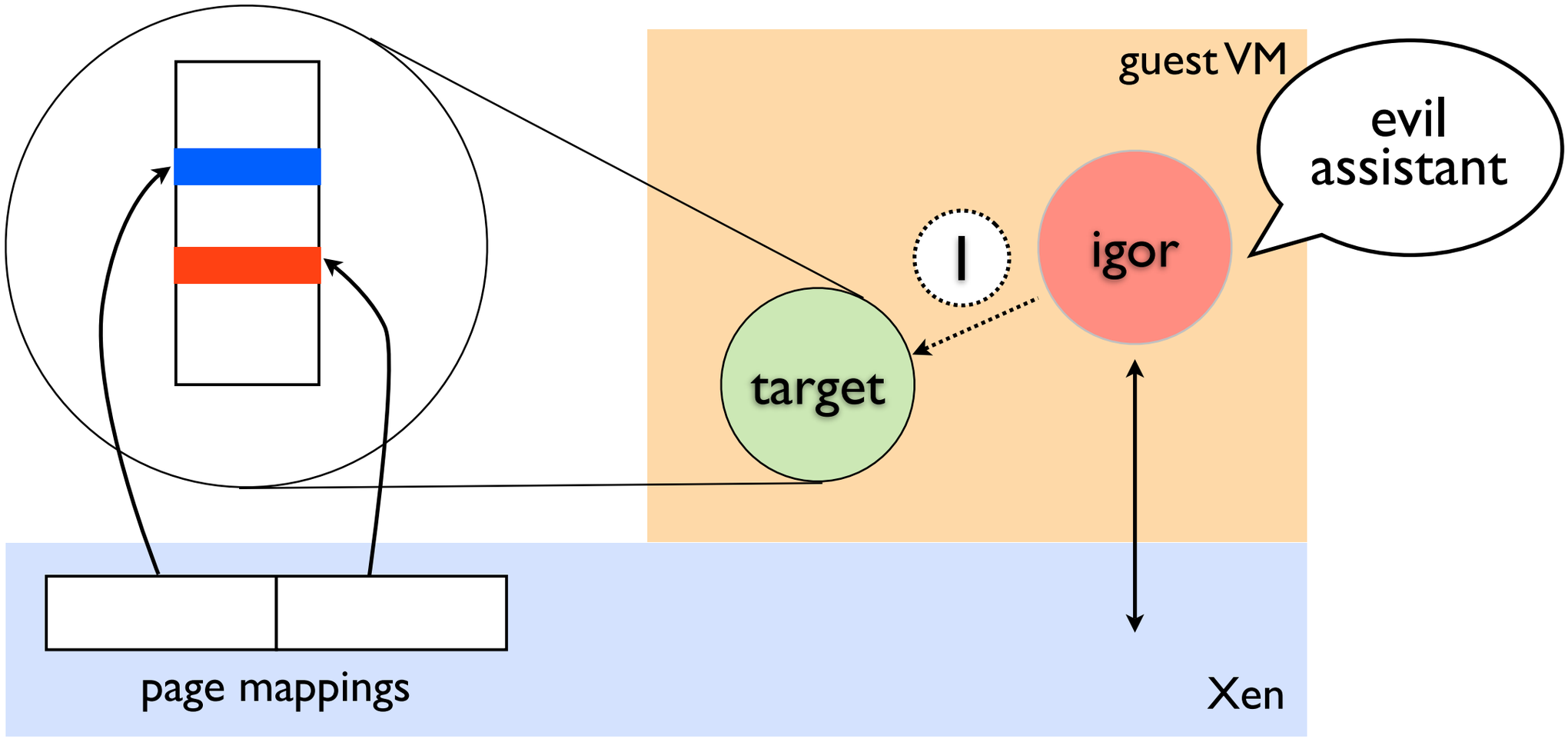
VMM + user level system



VMM + user level system

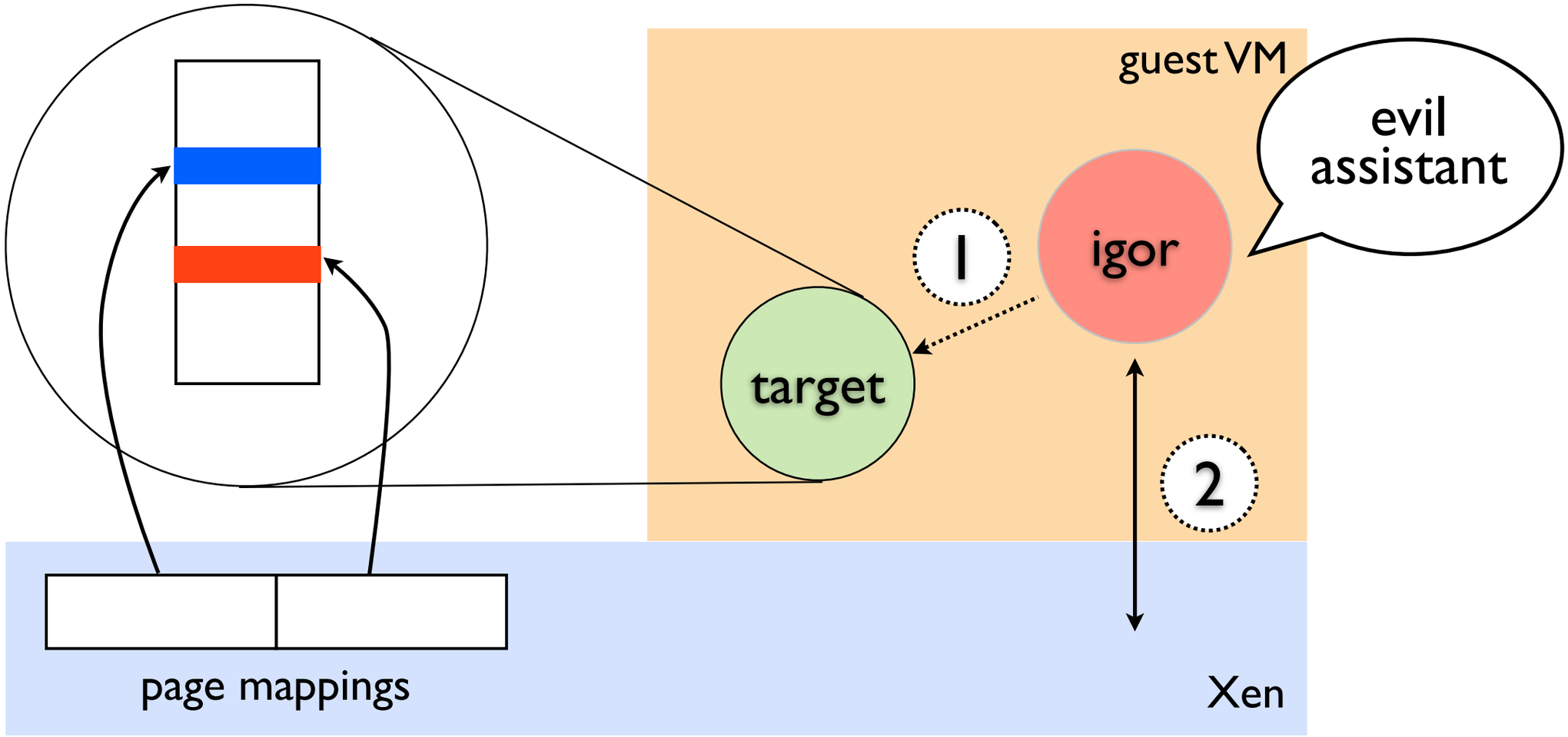


VMM + user level system



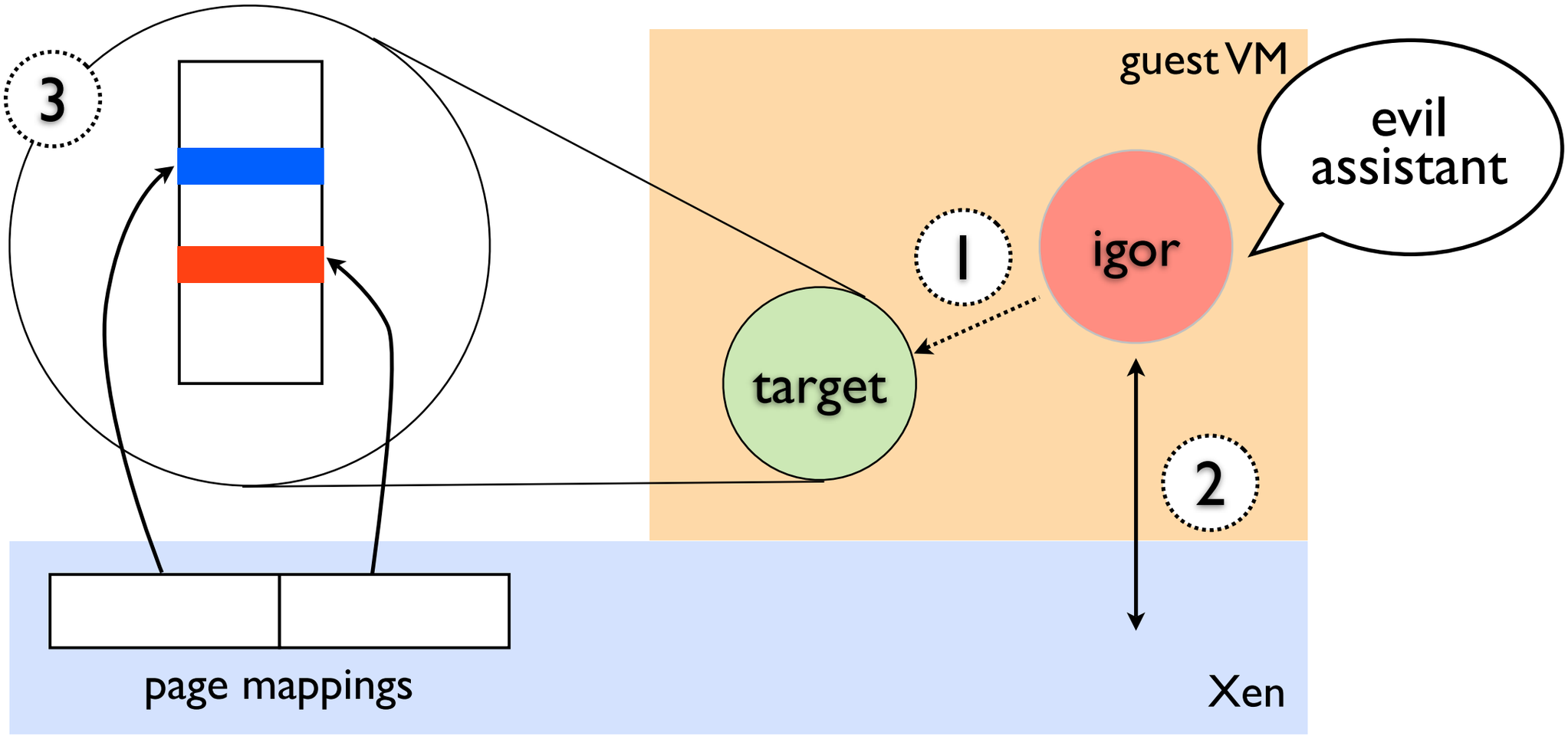
copy & modify pages

VMM + user level system



copy & modify pages ⟶ alert hypervisor

VMM + user level system



copy & modify pages → alert hypervisor → target continues *blissfully unaware*

How much does it cost?

Per MMU update

checking/enforcing protection bits

maintaining mapping structures

normal pages + 8%

context sensitive pages + 11%

Per Page Fault

checking for mapping

inducing TLB split

normal pages + 13%

context sensitive pages + 132%

How much does it cost?

Per MMU update

checking/enforcing protection bits

maintaining mapping structures

normal pages + 8%

context sensitive pages + 11%

Per Page Fault

checking fo
mapping

[relatively] complex

inducing TLB split

normal pages + 13%

ntext sensitive pages + 132%

Other uses

OS tamper resistance

protecting system call tables

Shielded processes

protecting processes from
malicious privileged access