Lecture-6 (Cache Attacks)
CS665-Fall 2018
Secure Memory Systems

Biswa@CSE-IITK

Before That: Bit of Crypto (Information)

$x \leftarrow 1$

for $i \leftarrow |e|\text{-}1$ **downto** 0 **do**

$x \leftarrow x^2 \bmod n$

if $(e_i = 1)$ **then**

$x = xb \bmod n$

endif

done

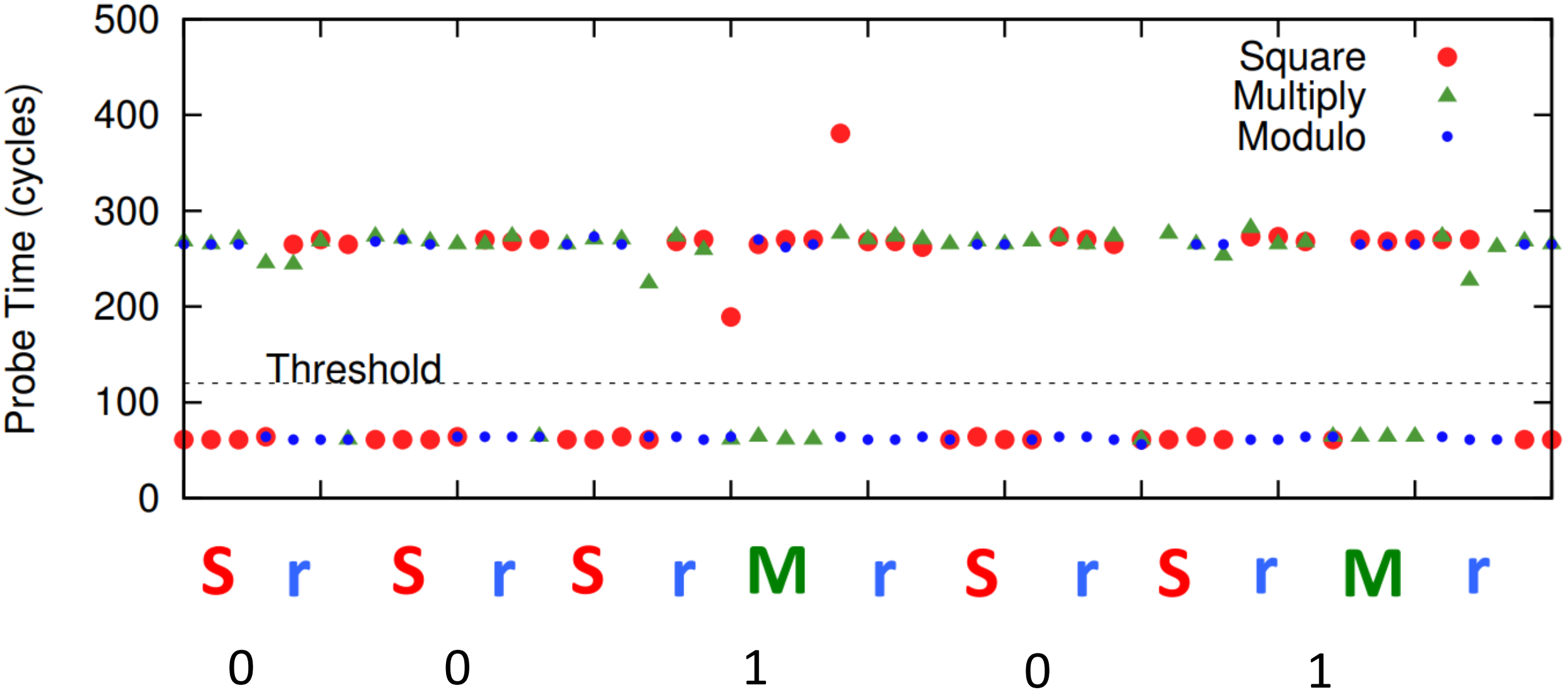
return x

Modular exponentiation, b^e mod n

Exponent e is used for decryption

Your password ~ Exponent e

Then


Probe Time (cycles)
500
400
300
200
100
0
Square
Multiply
Modulo
Threshold
S r S r S r M r S r S r M r
0 0 1 0 1

Flush (Evict) + Reload

LLC

Step 0:Spy *maps* the shared library, shared in the cache

Flush + Reload

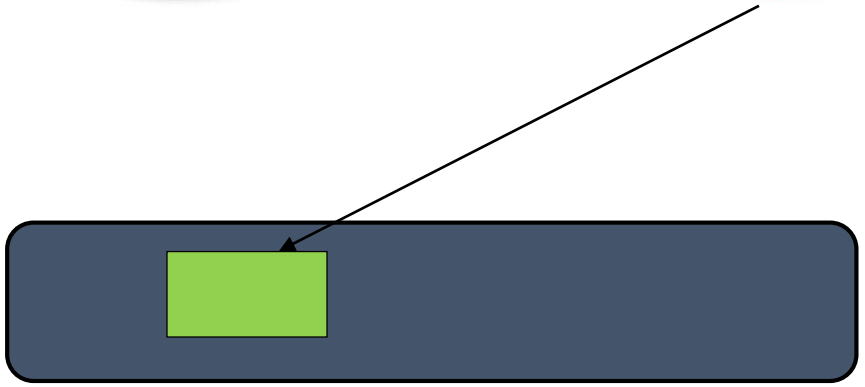
Clflush

LLC

Step 0:Spy *maps* the shared library, shared in the cache

Step 1:Spy *flushes* the cache block

Flush + Reload



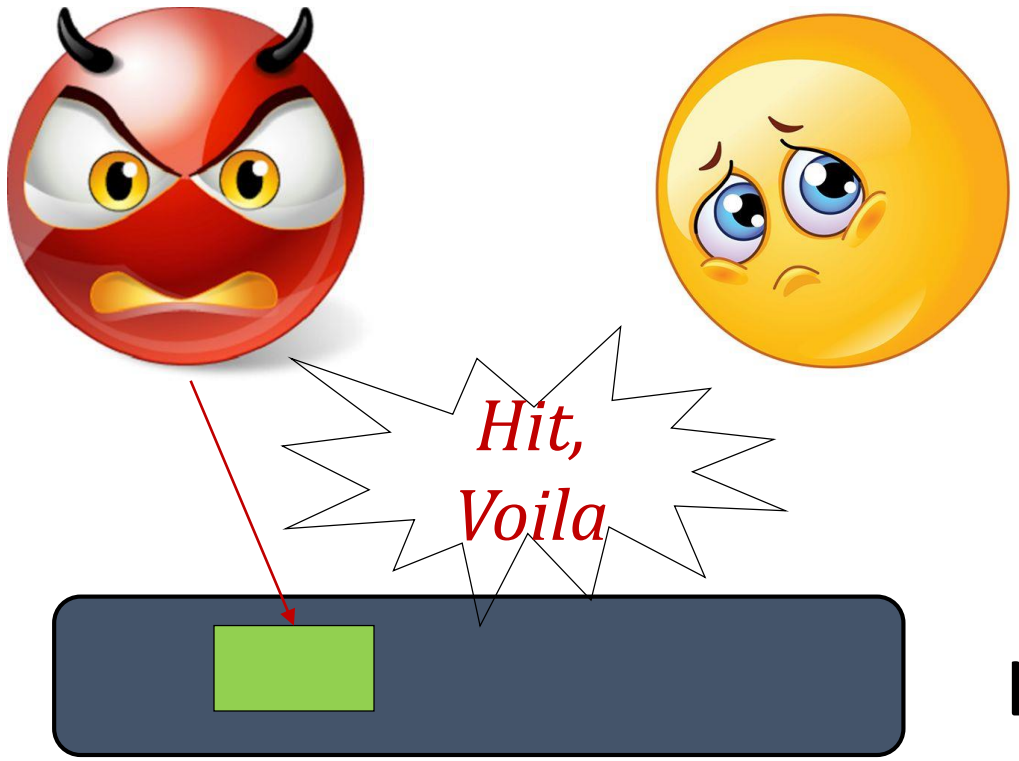
LLC

Step 0:Spy *maps* the shared library, shared in the cache

Step 1:Spy *flushes* the cache block

Step 2: Victim *reloads* the cache block

Flush + Reload



LLC

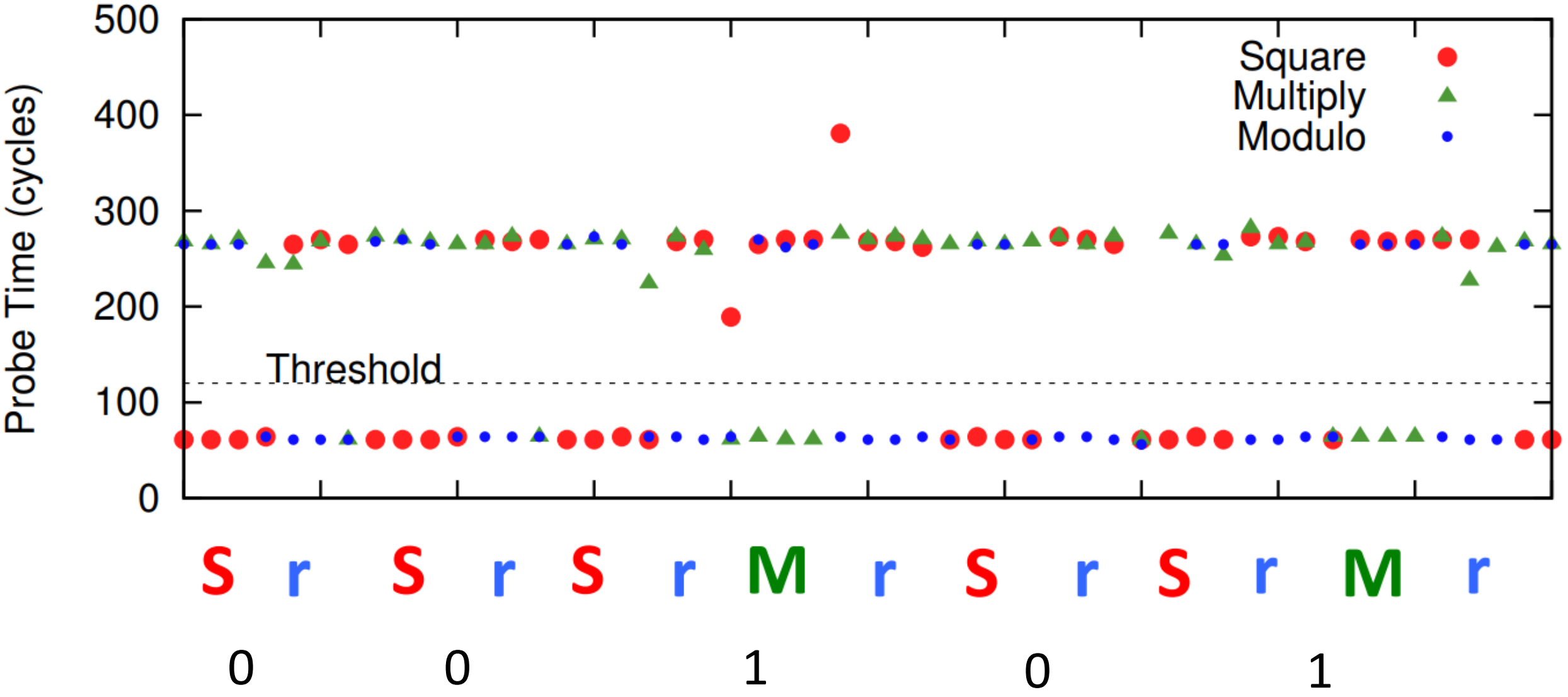
Step 0:Spy *maps* the shared library, shared in the cache

Step 1:Spy *flushes* the cache block

Step 2: Victim *reloads* the cache block

Step 3: Spy *reloads* the cache block (hit/miss)

What are the Cache Block Addresses?



Flush + Flush

LLC

Step 0:Spy *maps* the shared library, shared in the cache

Flush + Flush

Clflush

LLC

Step 0:Spy *maps* the shared library, shared in the cache

Step 1:Spy *flushes* the cache block

Flush + Flush

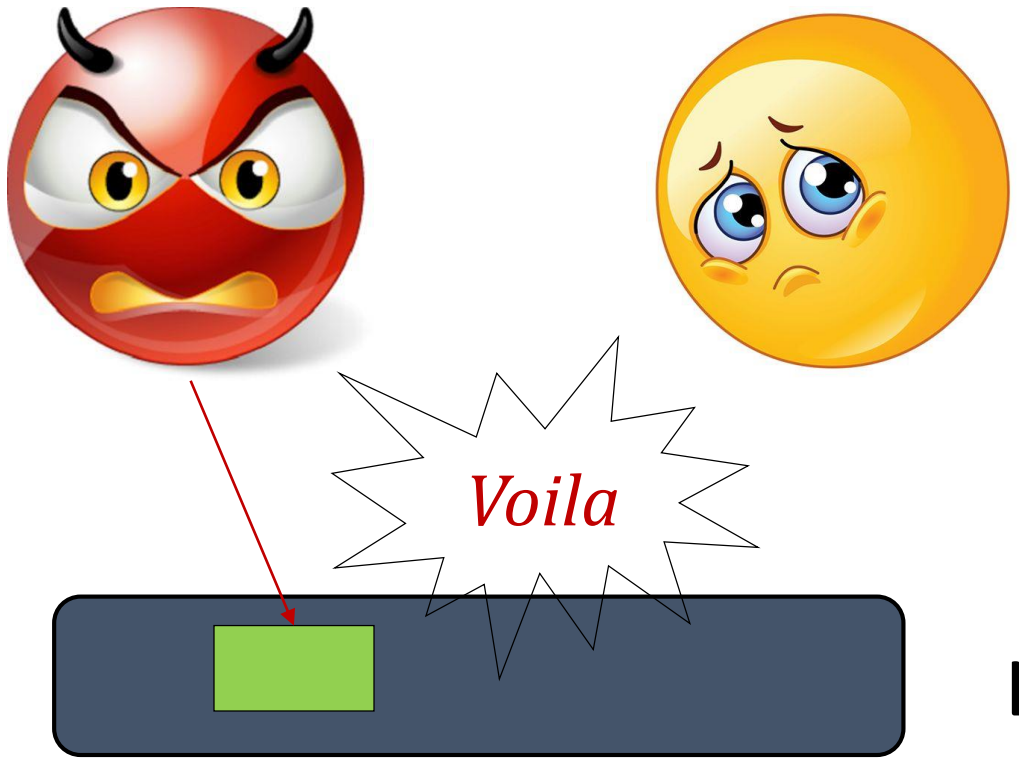
LLC

Step 0:Spy *maps* the shared library, shared in the cache

Step 1:Spy *flushes* the cache block

Step 2: Victim *reloads* the cache block

Flush + Flush



Step 0:Spy *maps* the shared library, shared in the cache

Step 1:Spy *flushes* the cache block

Step 2: Victim *reloads* the cache block

Step 3: Spy *flushes* the cache block again

Confused?

Clflush

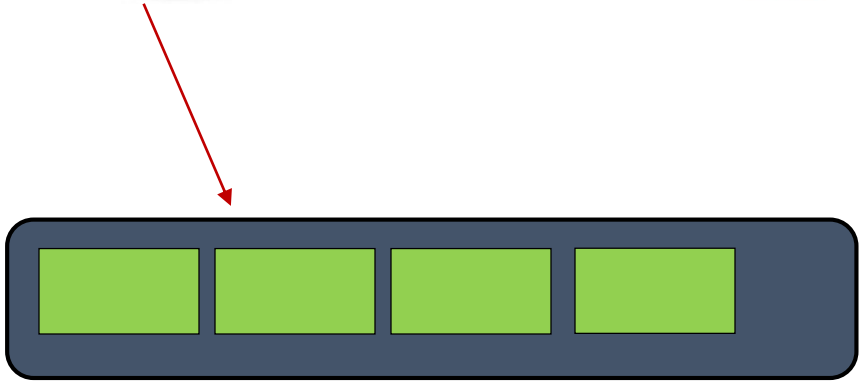
On a hit at the LLC, clflush has to flush L1+L2 of the victim too

On a miss at the LLC, do nothing (faster than hits)

Clflush [DIMVA '16]

Prime + Probe

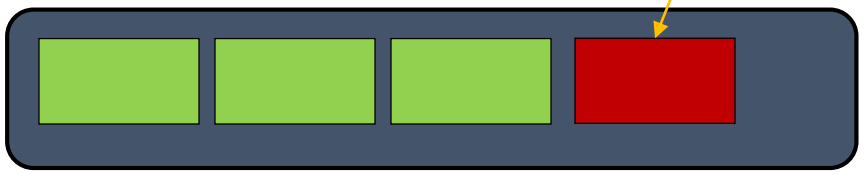
Prime + Probe



LLC

Step 0:Spy *fills* the entire shared cache

Prime + Probe

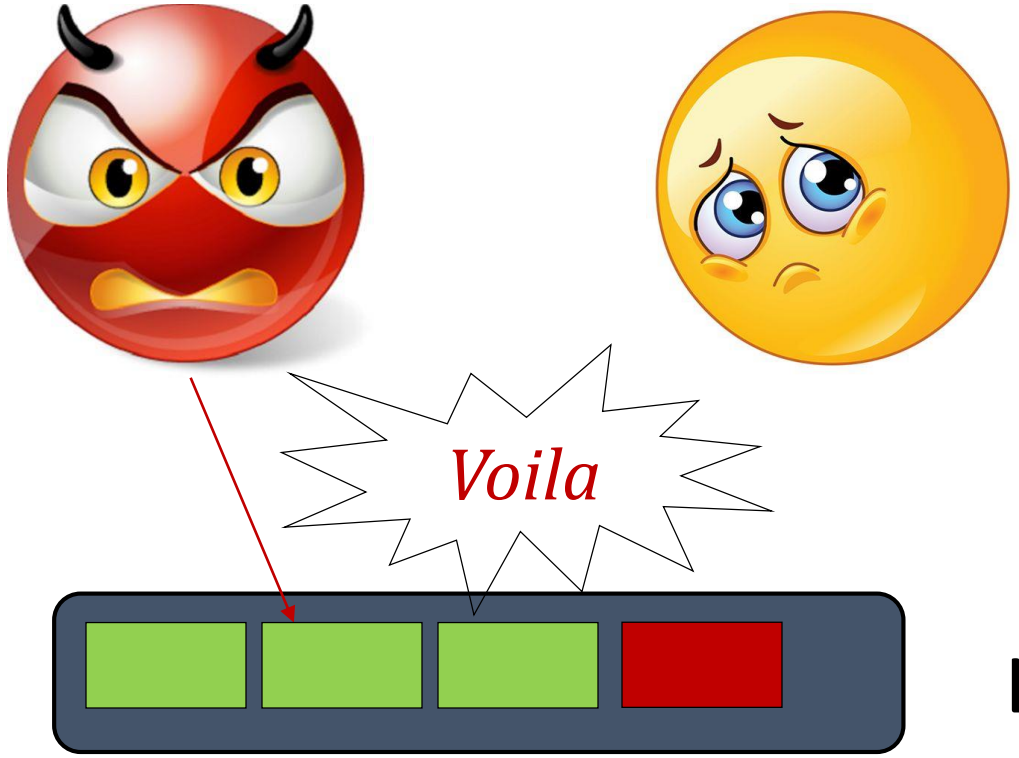


LLC

Step 0:Spy *fills* the entire shared cache (sets)

Step 1: Victim *evicts* cache blocks while running

Prime + Probe



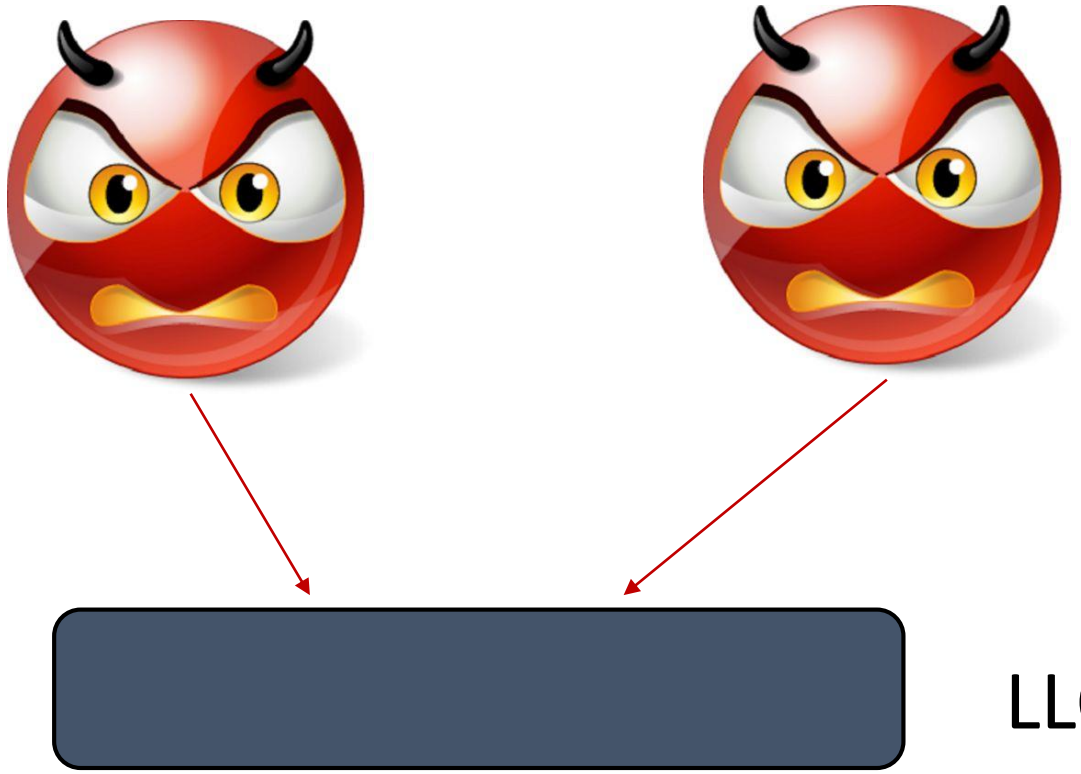
Step 0:Spy *fills* the entire shared cache

Step 1: Victim *evicts* cache blocks while running

Step 2: Spy *probes* the cache set

If misses then victim has accessed the set

Covert Channel



LLC

Step 0: Receiver gets data from L1 (fast, bit “0”)

Step 1: Sender thrashes LLC and back-invalidates L1

Step 2: Receiver gets data from DRAM (slow, bit “1”)

The Subtle Issues

- Flush + Reload: Demands Page Sharing, Fine-grained, low-noise

(variant is flush+flush)

- Evict + Reload: An alternative for Flush + Reload, Almost equally effective (variant is Evict + Time)

- Prime + Probe: Does not demand page sharing, coarse grained, high-noise, need to find out the eviction set

More Subtle Issues: Flush + Reload

- Works across CPU sockets
- Works on non-inclusive caches
- However, it can only recover statically allocated data

More Subtle Issues: Evict + Reload

- No flush instruction
- Attacker can use huge pages
- Applicable to processors without clflush instruction
- Only work with inclusive caches in the same CPU socket
- Need information about LLC slices

Subtle Issues with Prime+Probe

- Need to create an eviction set(s): Refer “LLC side-channels are practical”

- LLC slicing information is needed: Need to reverse engineer

The Notion of Time

- Gap between Evict and Reload, Flush and Reload, Prime and Probe

- A subtle parameter for a successful attack