Exzess: Hardware-based RAM Encryption against Physical Memory Disclosure

Alexander Würstlein Michael Gernoth
Johannes Götzfried Tilo Müller

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

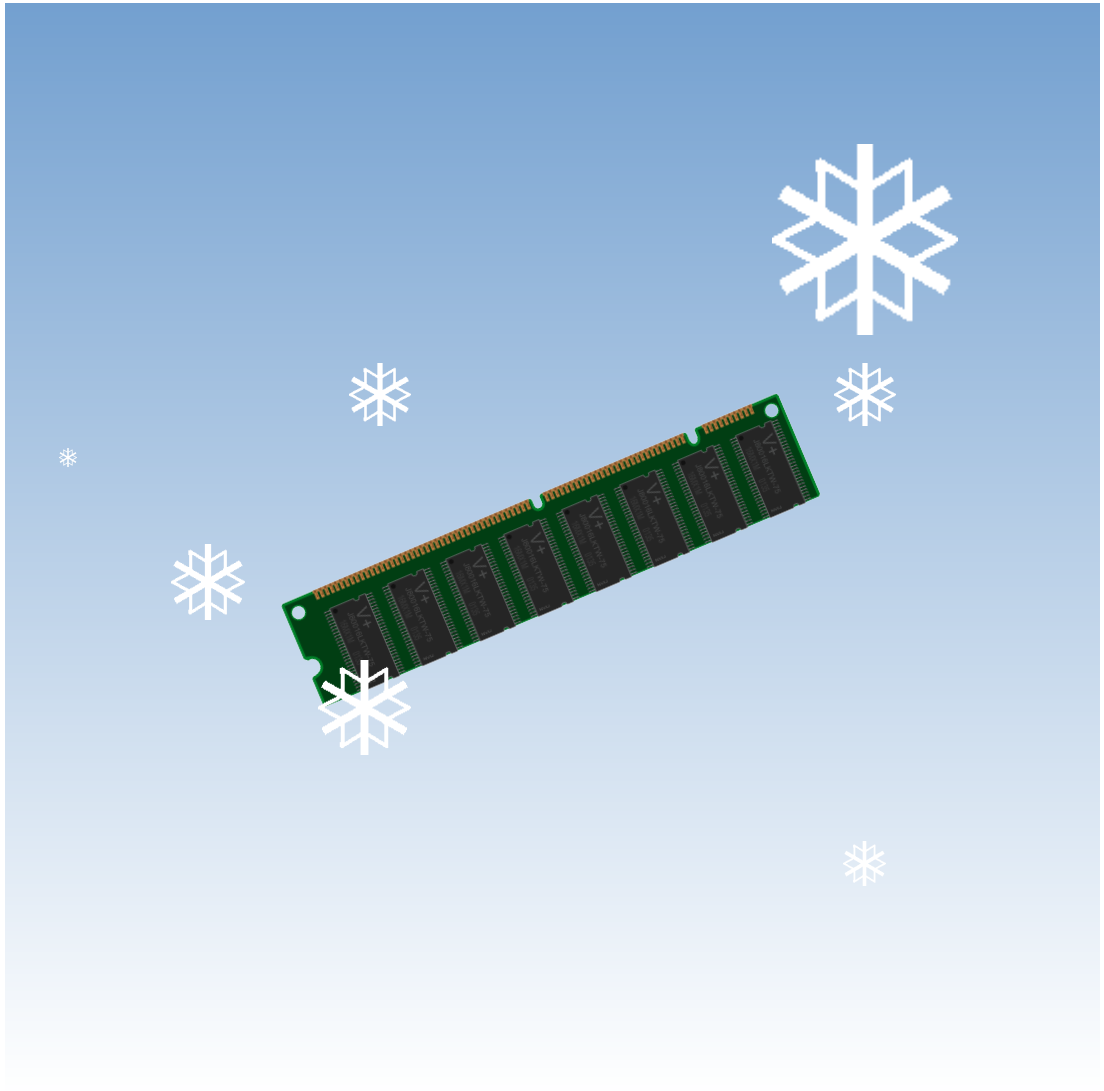
DFG Deutsche Forschungsgemeinschaft

This work was partly supported by the German Research Foundation (DFG) as part of the Transregional Collaborative Research Centre "Invasive Computing" (SFB/TR 89).

2016-04-05

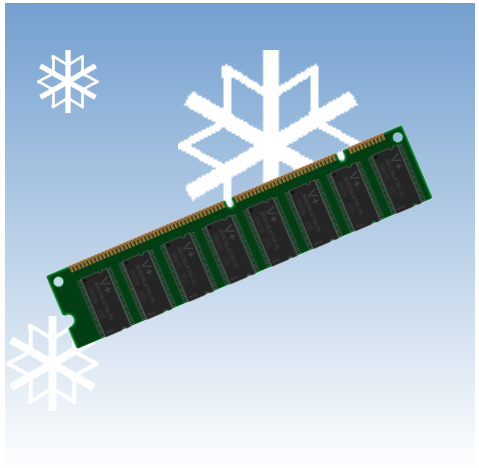
The Problem

The Problem: Cold Boot Attack



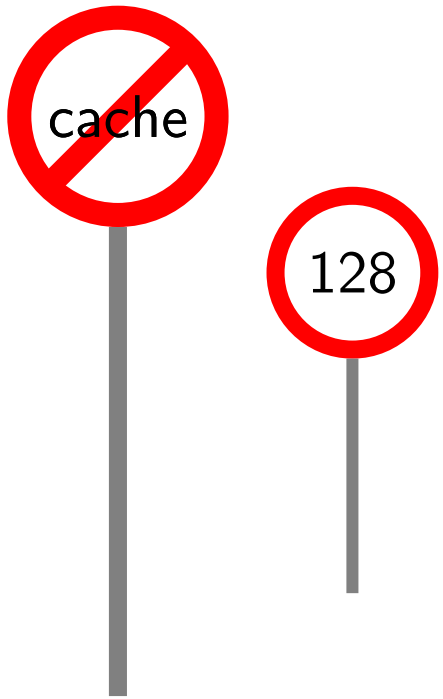
Physical Memory Disclosure

- remanence effect
  - DRAM contents readable after power-down
  - for a few minutes
- RAM modules
  1. cooled down
  2. physically extracted
  3. contents copied
- endangers:
  - in-use cryptographic keys (X.509, ssh)
  - passwords
  - full-disk-encryption keys
  - other sensitive data



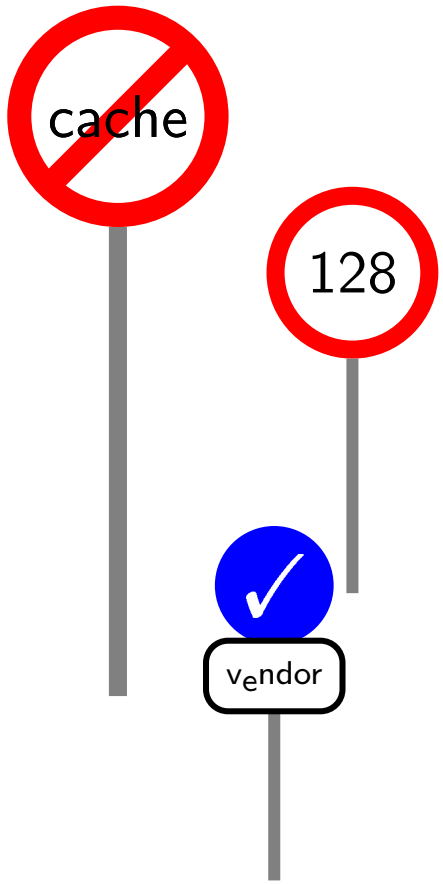
Solutions

- software-based attempts
  - slow:
    - disable caches
    - expensively encrypt everything
  - limited:
    - only protect one 128bit key
    - only apply to certain types of secrets (e.g. FDE keys)
  - incompatible:
    - require extensive software changes



Solutions

- software-based attempts
  - slow:
    - disable caches
    - expensively encrypt everything
  - limited:
    - only protect one 128bit key
    - only apply to certain types of secrets (e.g. FDE keys)
  - incompatible:
    - require extensive software changes
- hardware-based attempts
  - embedded-only
  - limited scope
    - only for code signed & blessed by third party



Our Proposed Solution

Goals

- work transparently
- multiple uses
- be fast
- hardened hardware

Means

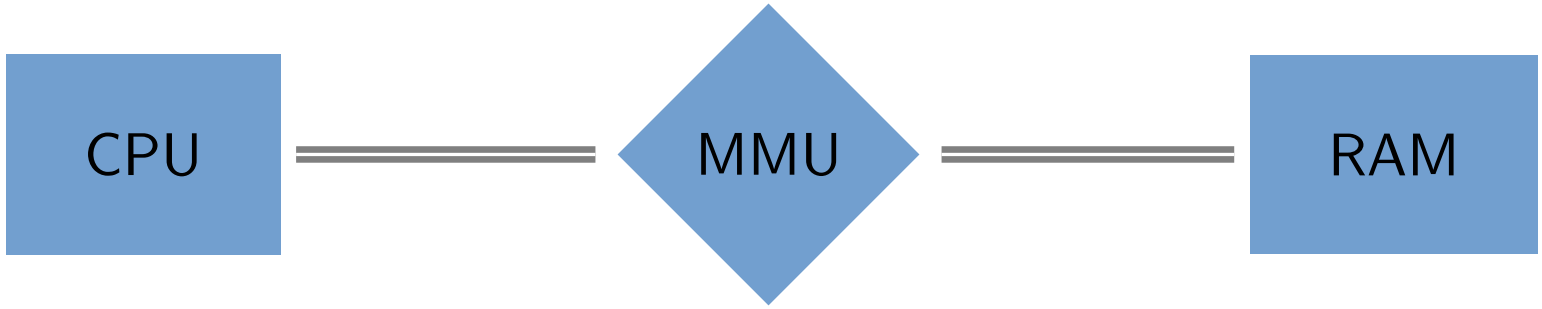
$\Rightarrow$ memory as our interface

$\Rightarrow$ no painful size limitations

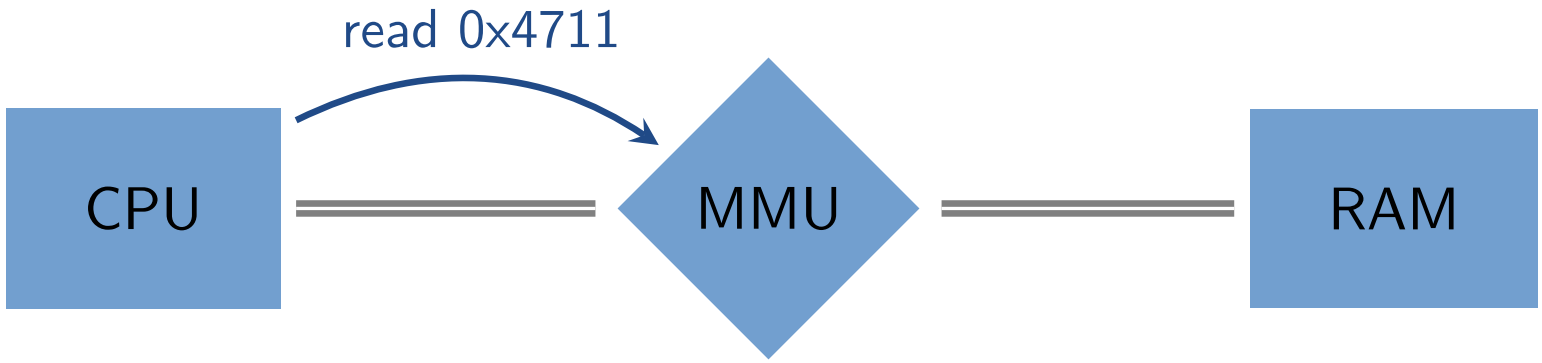
$\Rightarrow$ encrypt only important data

$\Rightarrow$ be tamper-proof & simple

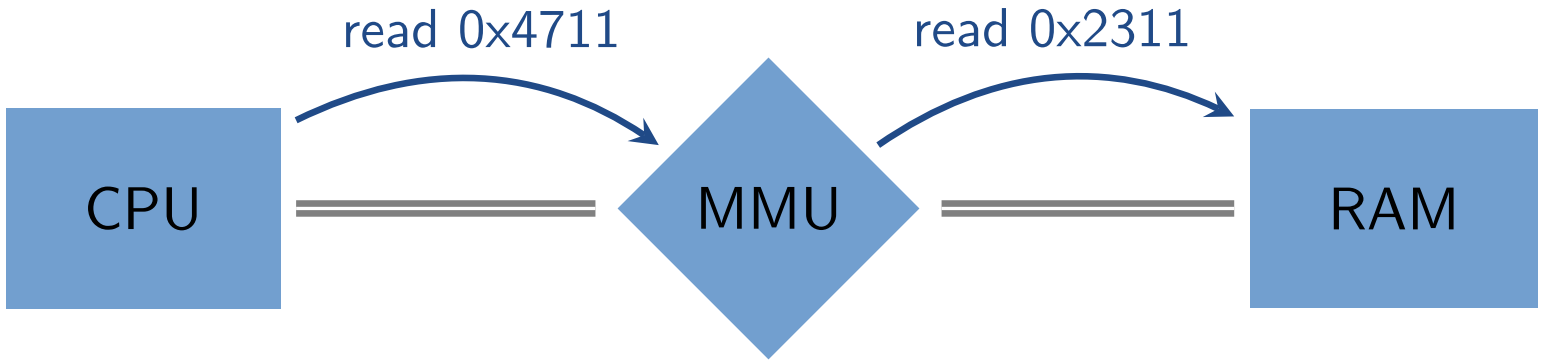
How?



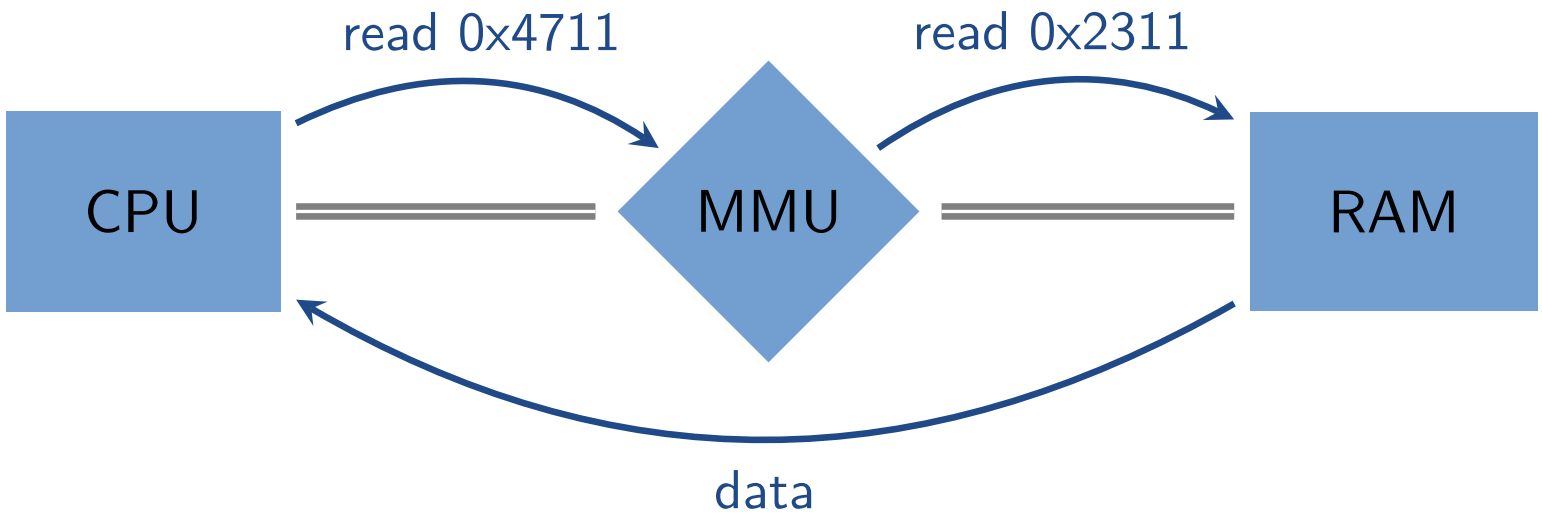
How?



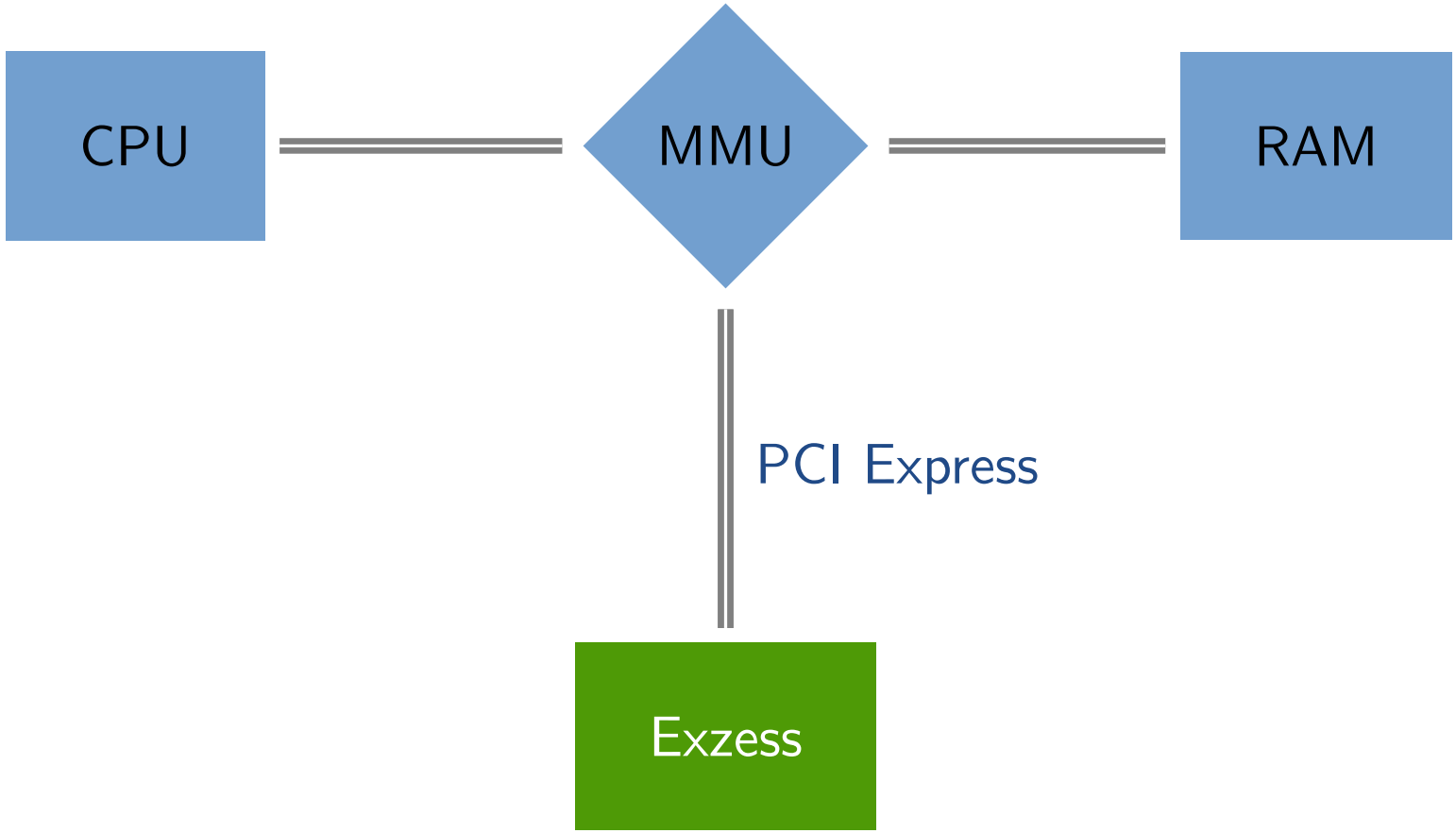
How?



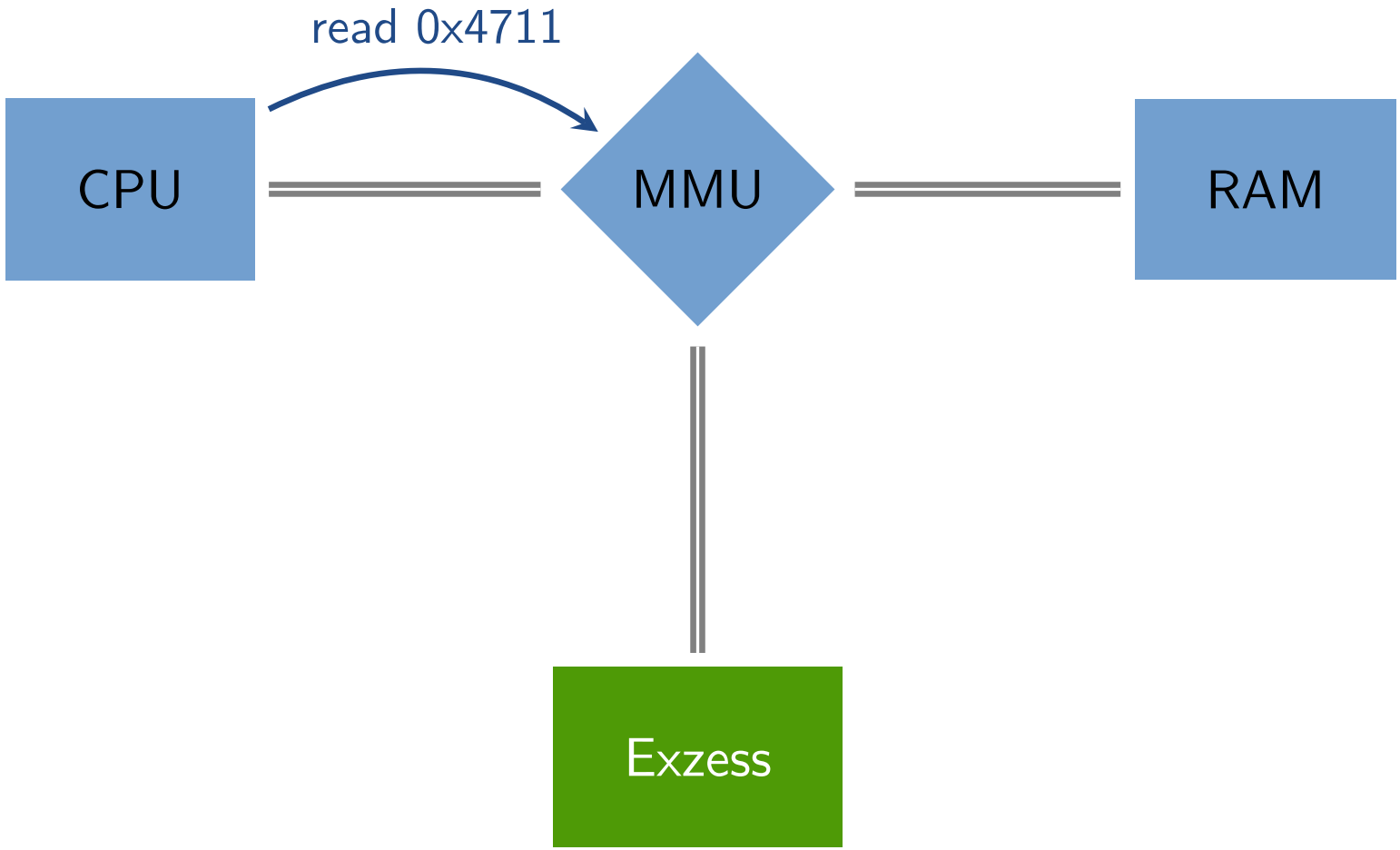
How?



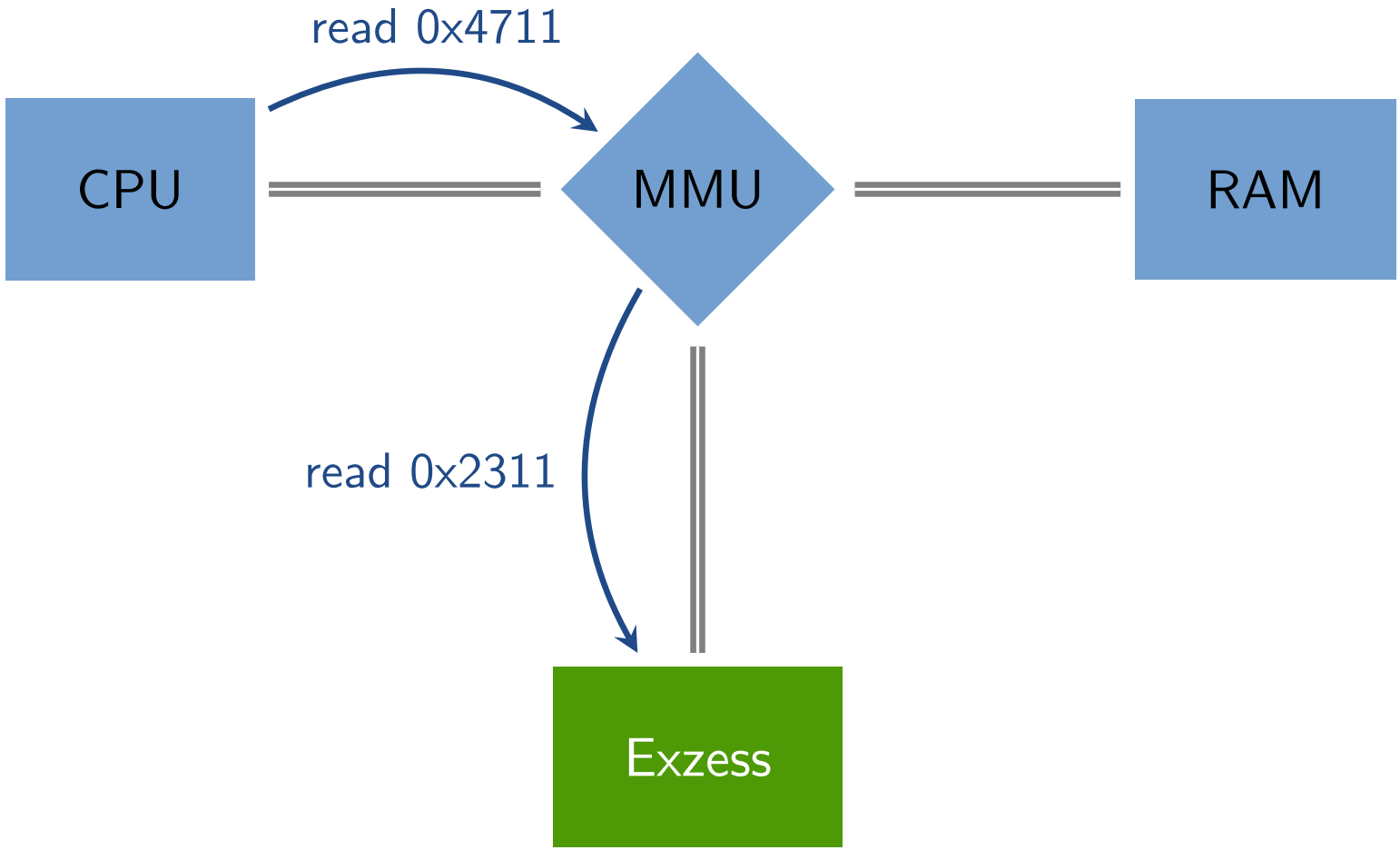
How?



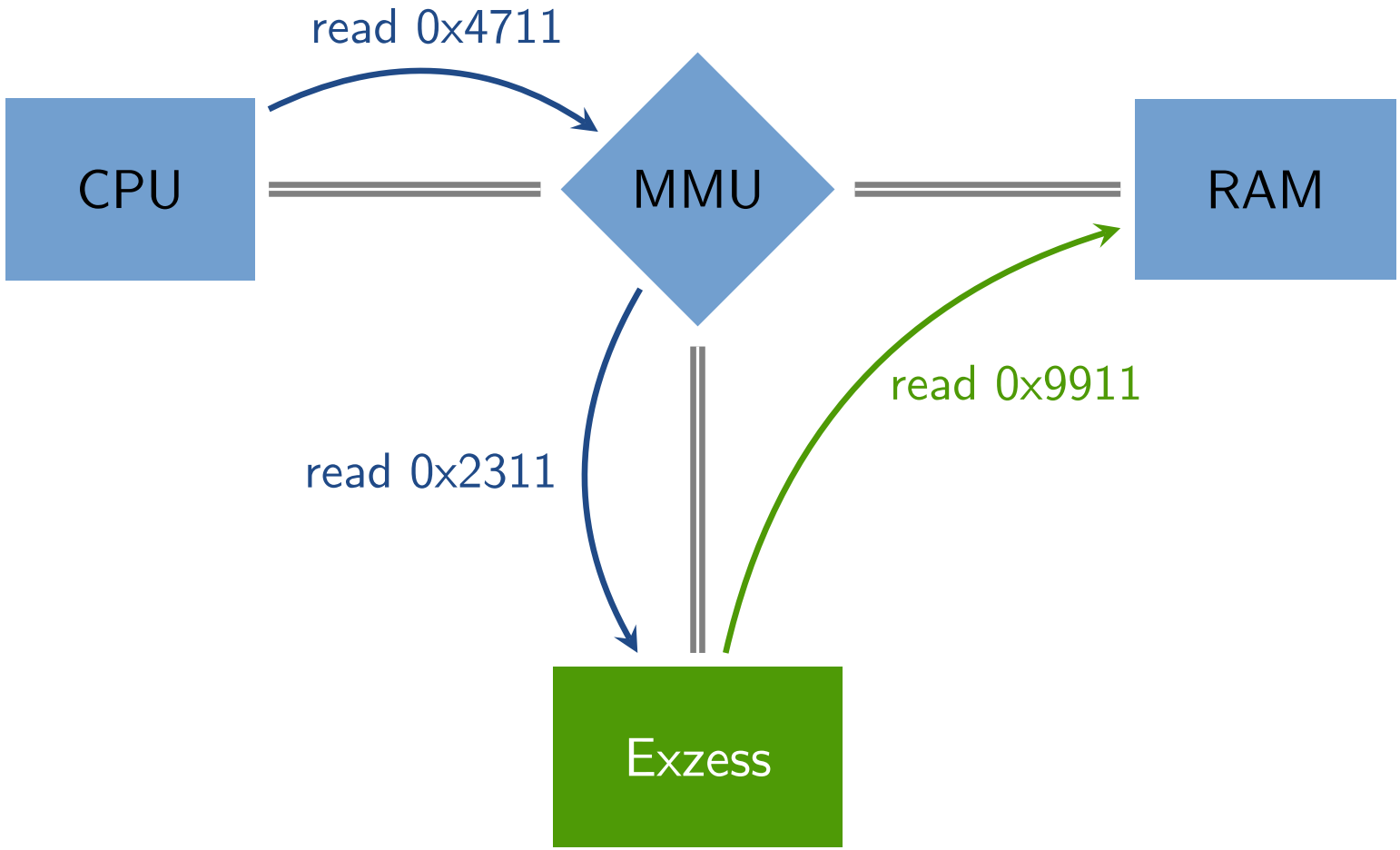
How?



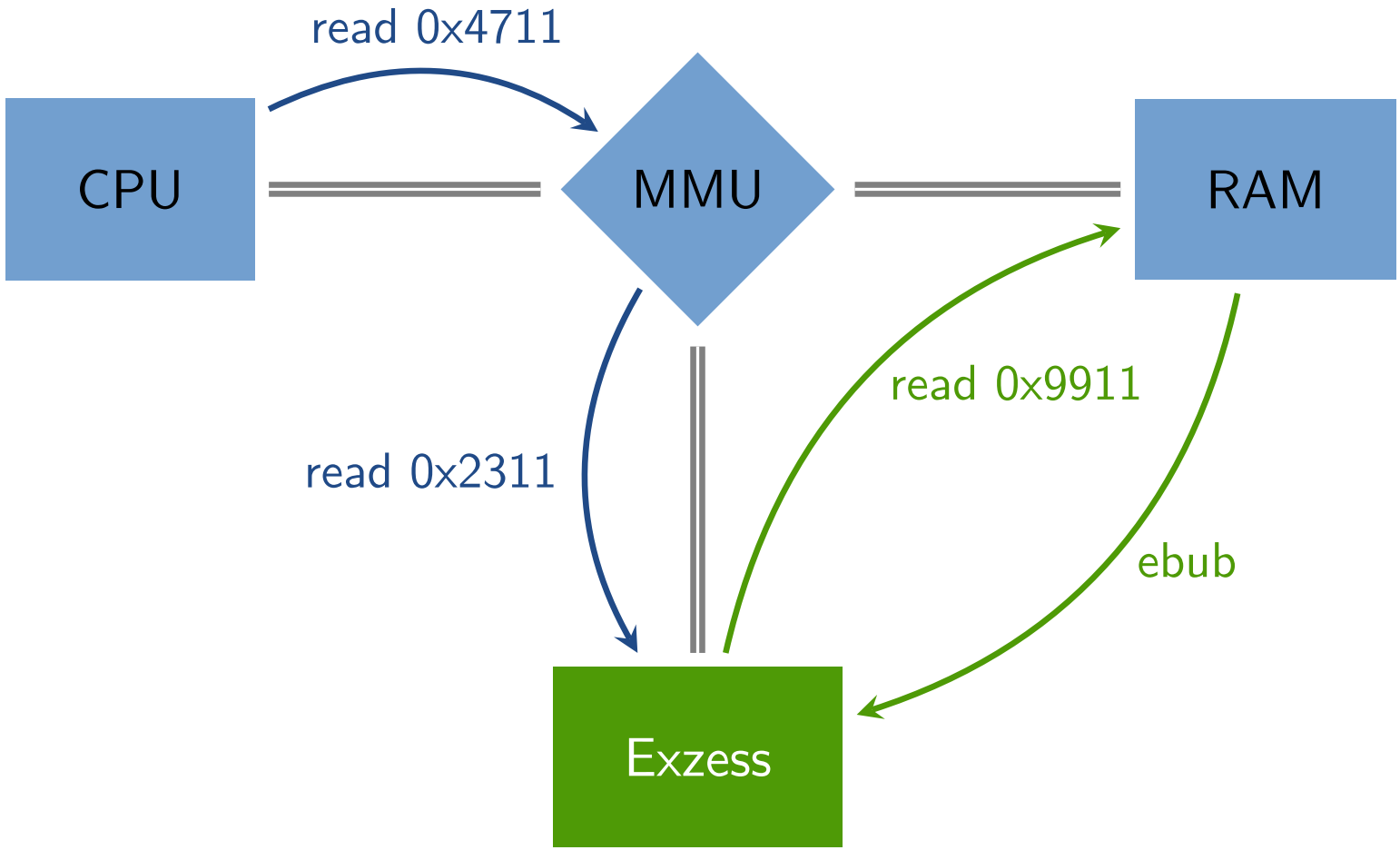
How?



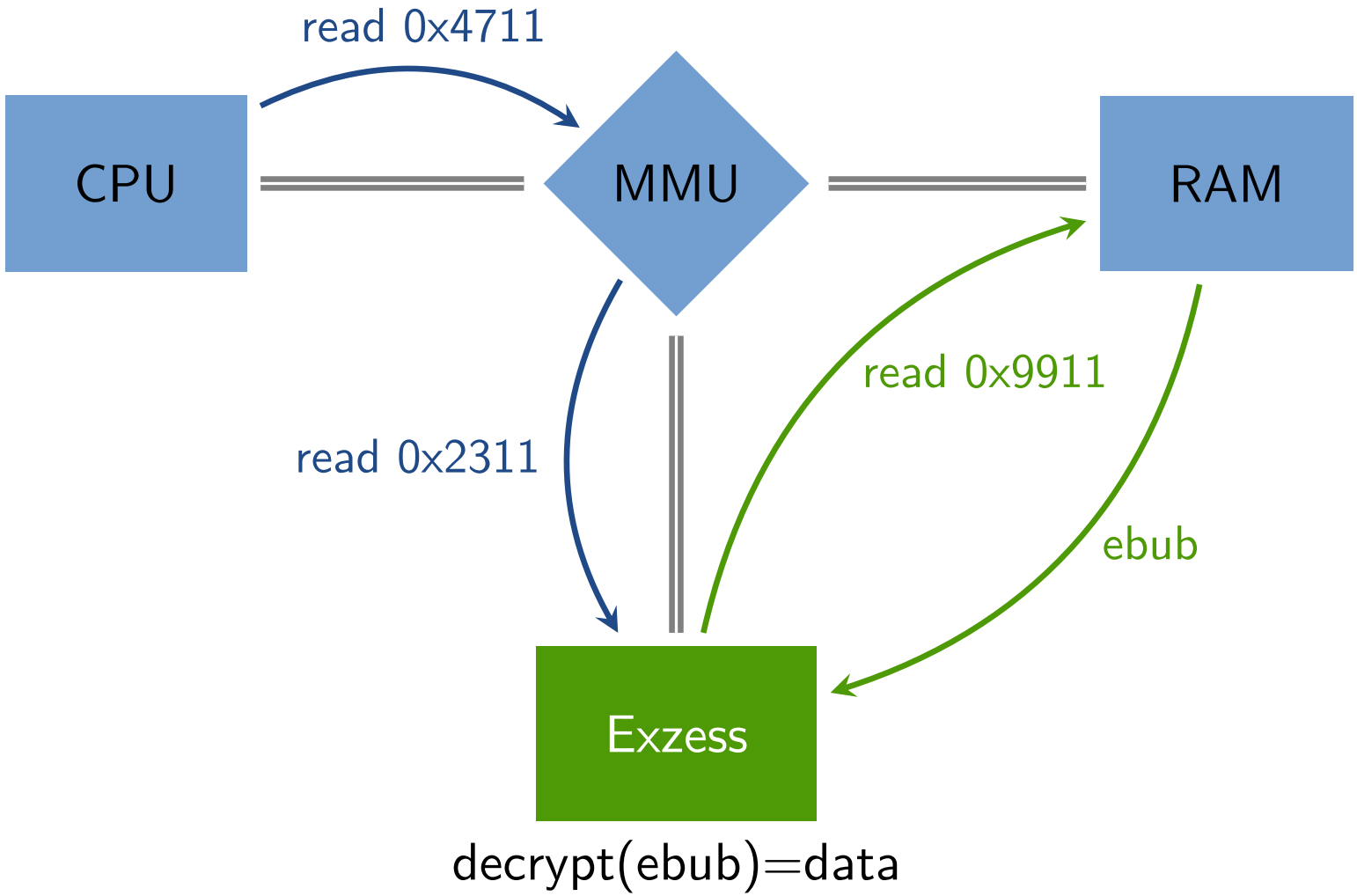
How?



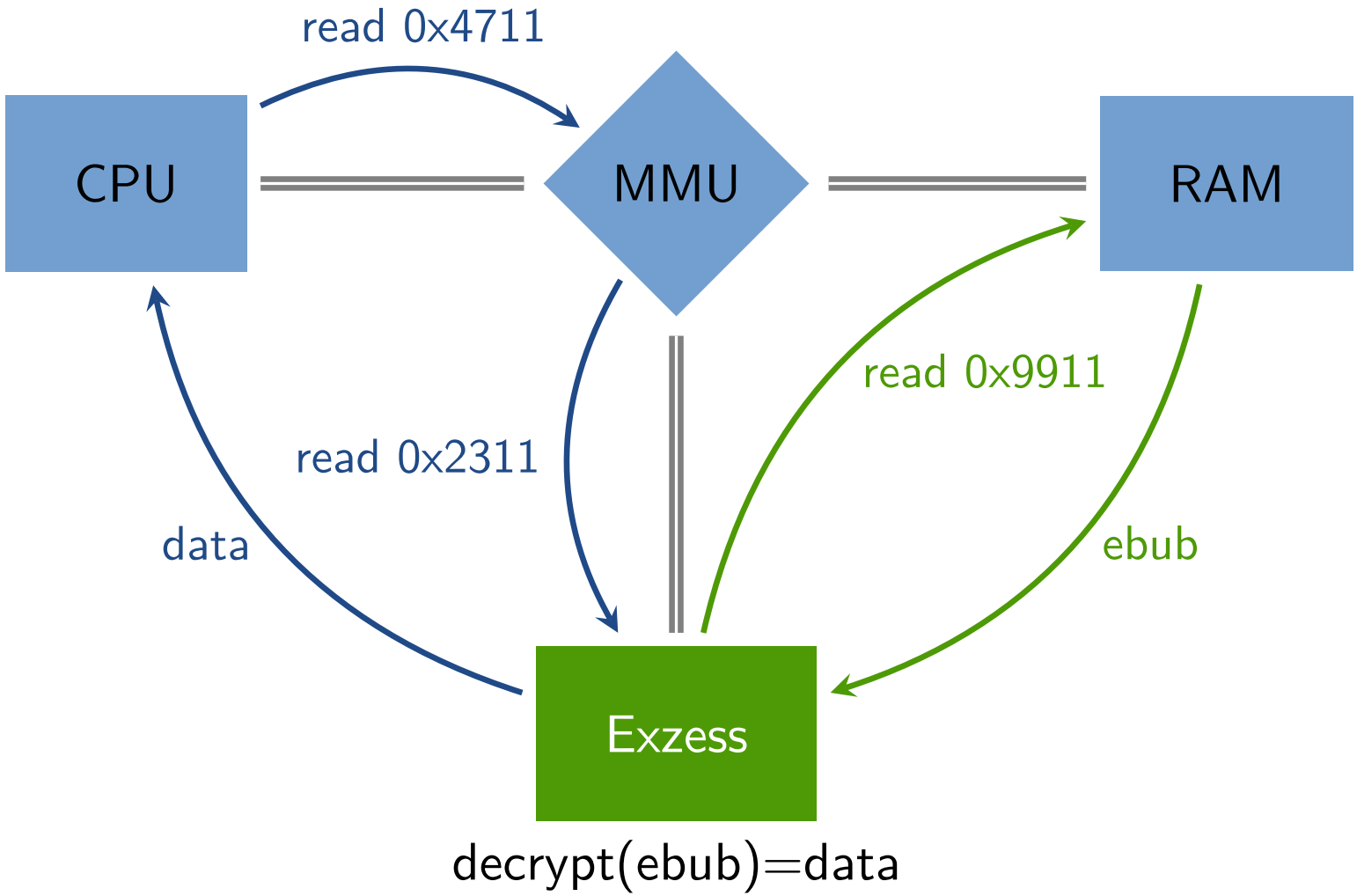
How?



How?

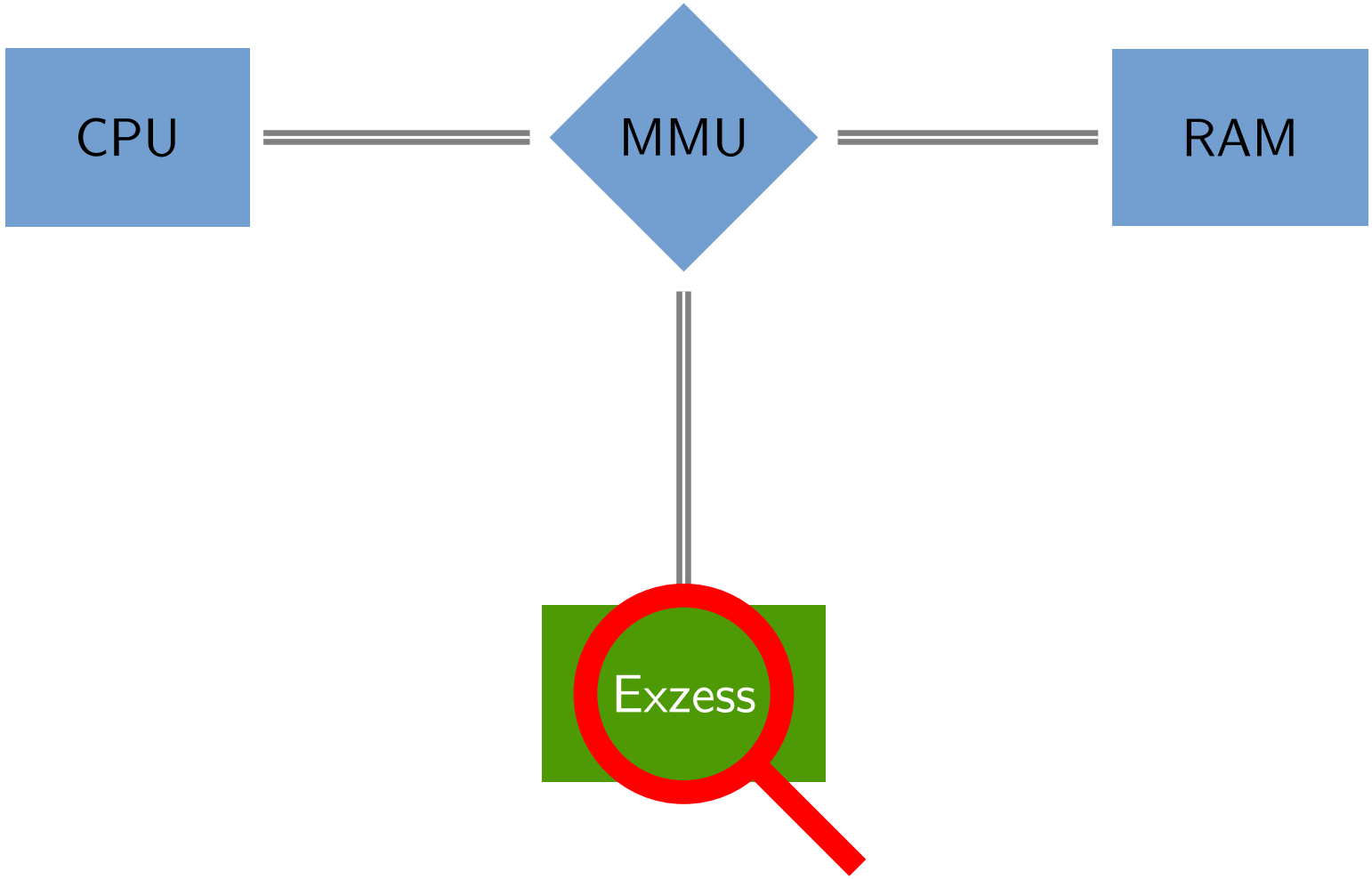

read 0x4711
CPU
MMU
RAM
read 0x9911
read 0x2311
ebub
Exzess
decrypt(ebub)=data

How?

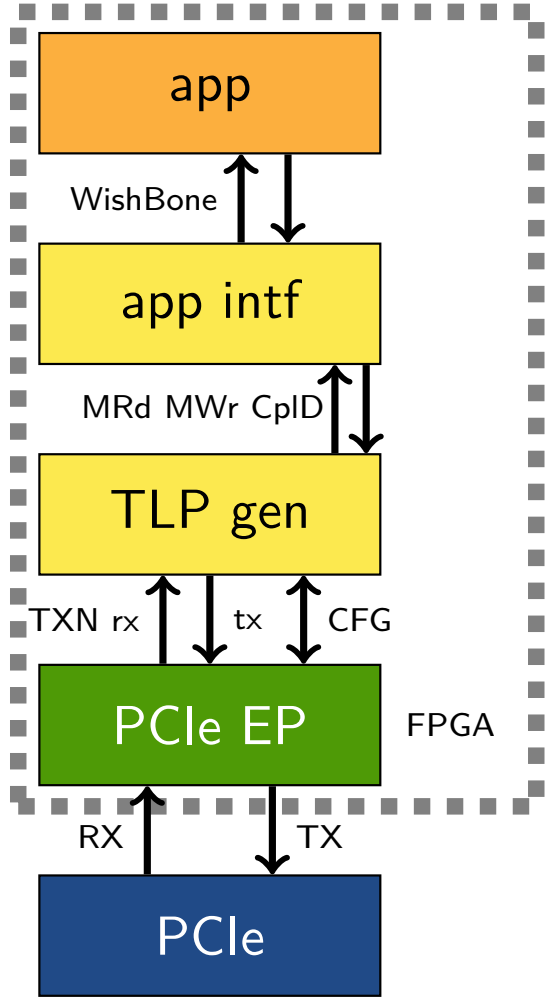


decrypt(ebub)=data

How?



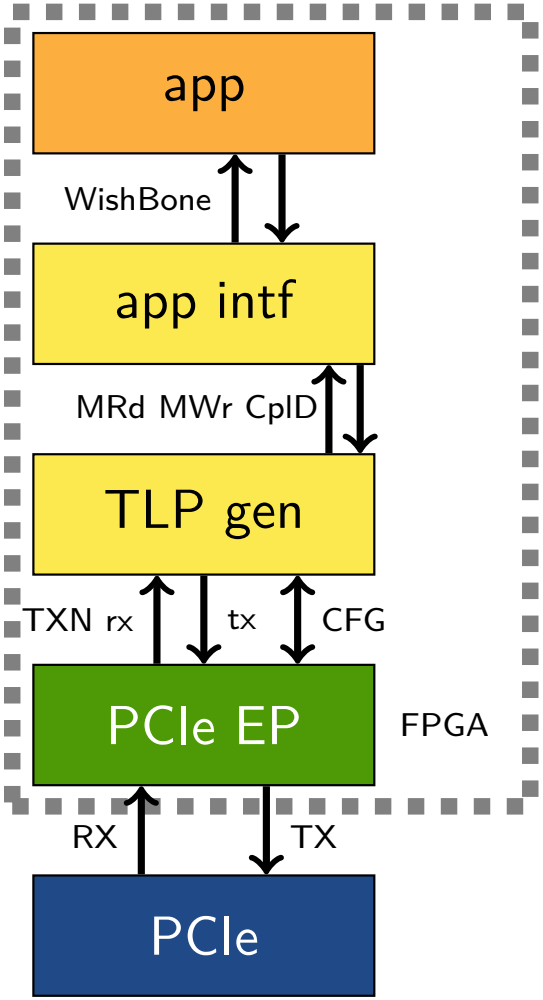
The Exzess FPGA



- PCI Endpoint
  - FPGA IP core
- PCI Express Bus

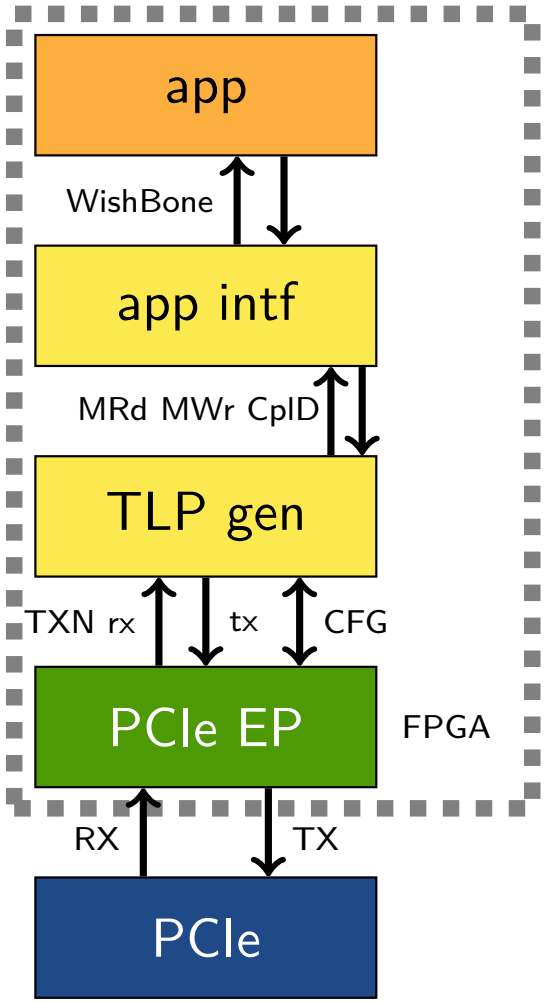
The Exzess FPGA

- application interface
  - Wishbone to TLP interface
- TLP generator
  - handles upper layers of PCIe protocol stack



The Exzess FPGA

- applications
  - XOR “encryption”
  - AES in CTR mode
  - others possible



Application View

- memory proxy
  - CPU issues write to memory
  - chipset redirects write to device
  - device encrypts data
  - device issues write to memory

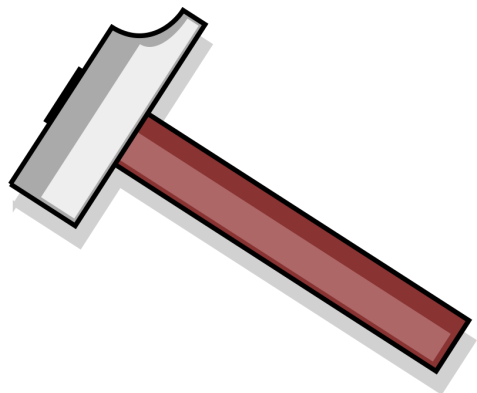
Application View

- memory proxy
  - CPU issues write to memory
  - chipset redirects write to device
  - device encrypts data
  - device issues write to memory
- usage
  - `mmap` the device memory
  - read & write as usual
  - Exzess will transparently ...
    - encrypt
    - decrypt
    - use normal RAM as encrypted backend storage

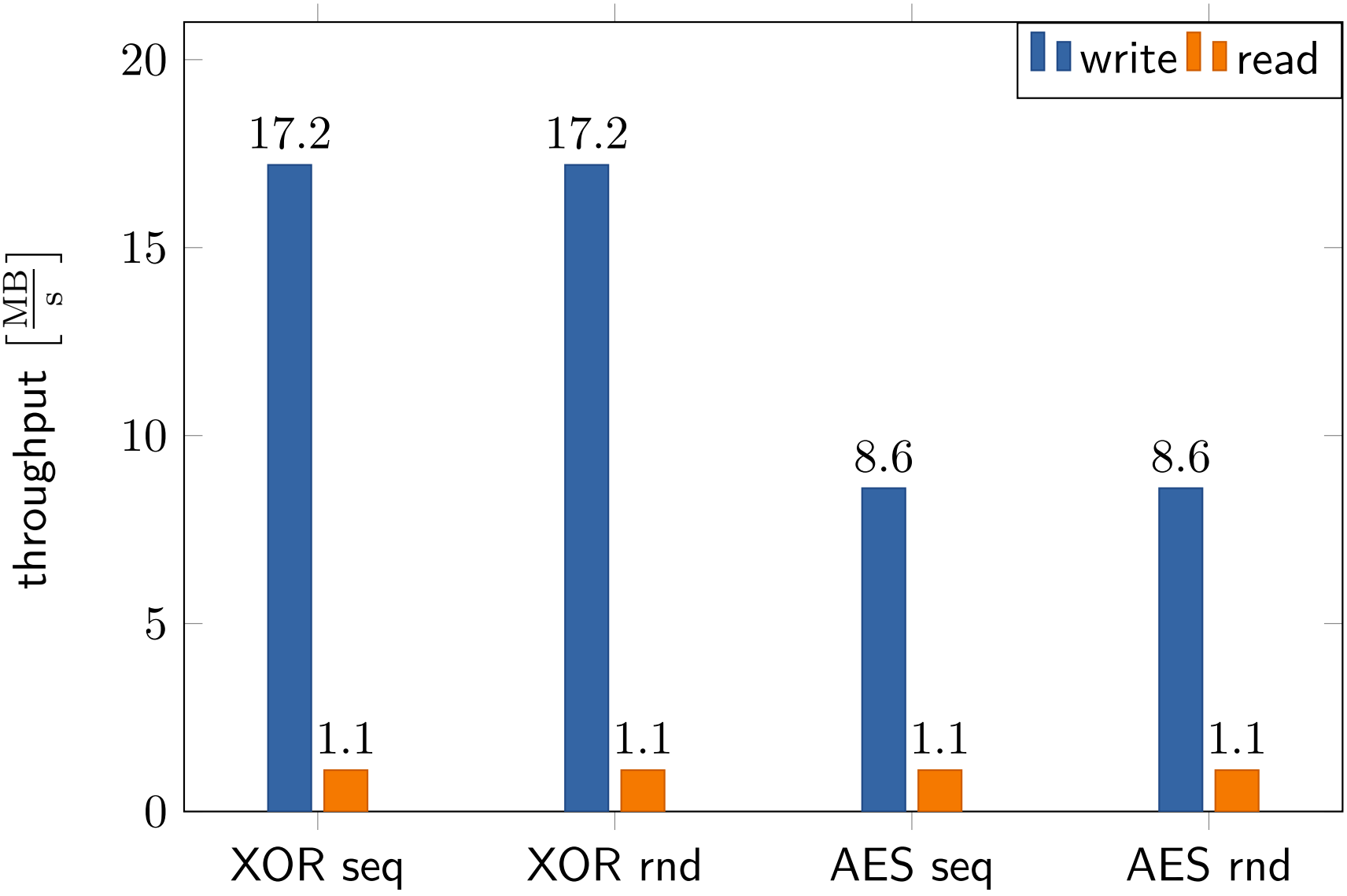
```
1  char *p = secure_malloc(6);
2
3  strcpy(p, "stuff");
4  do_anything(p);
5
6  secure_free(p);
```

Goals and Attacker Model

- data in physical RAM is encrypted
  - even searching `/dev/mem` is futile
  - only some data: non-secured applications without performance loss
  - large encryptable area: almost your whole RAM
  - RAM extraction useless: also extract & read key from Exzess
- improvements upon the prototype
  - FPGA might have: debug ports, emanations, ...
  - hardening and tamper-proofing: think HSM (Hardware Security Module)

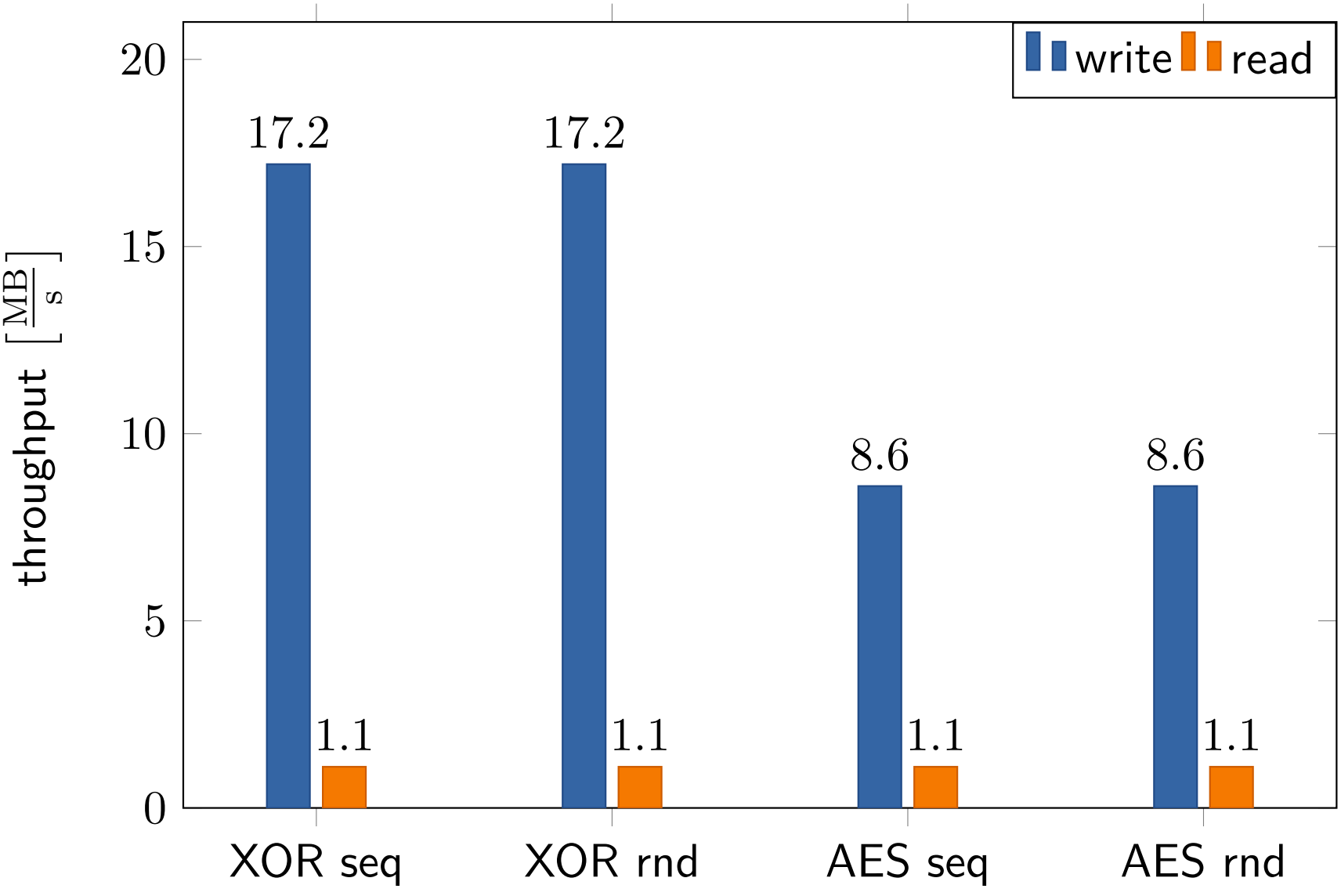


Performance Measurements of Prototype



Measurements on FPGA Prototype, not cached, 4-byte packets

Performance Measurements of Prototype



Measurements on FPGA Prototype, not cached, 4-byte packets

Future Work

- integrate into LUKS, OpenSSL, SSH,...
- hardened hardware
- PCIe firewalling
- beyond encryption: memory proxy application for...
  - debugging
  - data integrity checking
  - redundancy and healing

Conclusion

- Exzess is a working prototype
- can solve cold-boot problem
- interesting concept, even beyond encryption
- encrypting memory proxy
- source code: `https://www4.cs.fau.de/~arw/exzess/`