Introduction to Information Security

Side Channel

Sang Kil Cha

Side-Channel Attack

In cryptography, a side-channel attack is any attack based on information gained from the physical implementation of a cryptosystem, rather than brute force or theoretical weaknesses in the algorithms.

- From Wikipedia

Covert Channel

A covert channel is a type of computer security attack that creates a capability to transfer information objects between processes that are not supposed to be allowed to communicate by the computer security policy.

- From Wikipedia

Side-Channel vs. Covert Channel?

Unintended vs. Intended

Example: TCP Header

TCP header padding is used to ensure that the TCP header ends and data begins on a 32-bit boundary

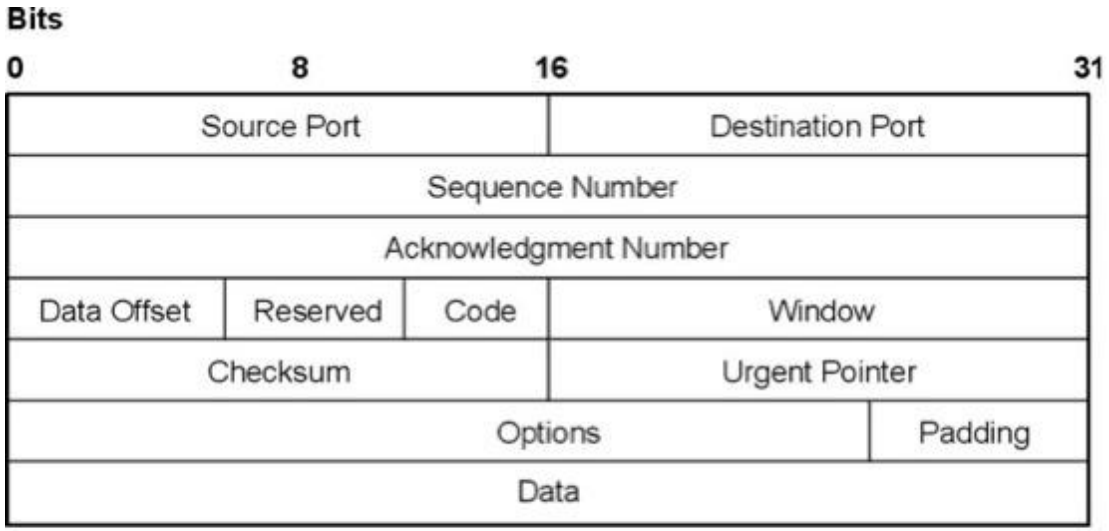


Image from http://ps-2.kev009.com/wisclibrary/aix51/usr/share/man/info/en_US/a_doc_lib/aixbman/commadmn/tcp_protocols.htm

Example: Electronic Voting

Image from https://arstechnica.com/tech-policy/2015/04/meet-the-e-voting-machine-so-easy-to-hack-it-will-take-your-breath-away/

Example: Eavesdropping

- Straightforward way: microphone under the table

- Side-channel?

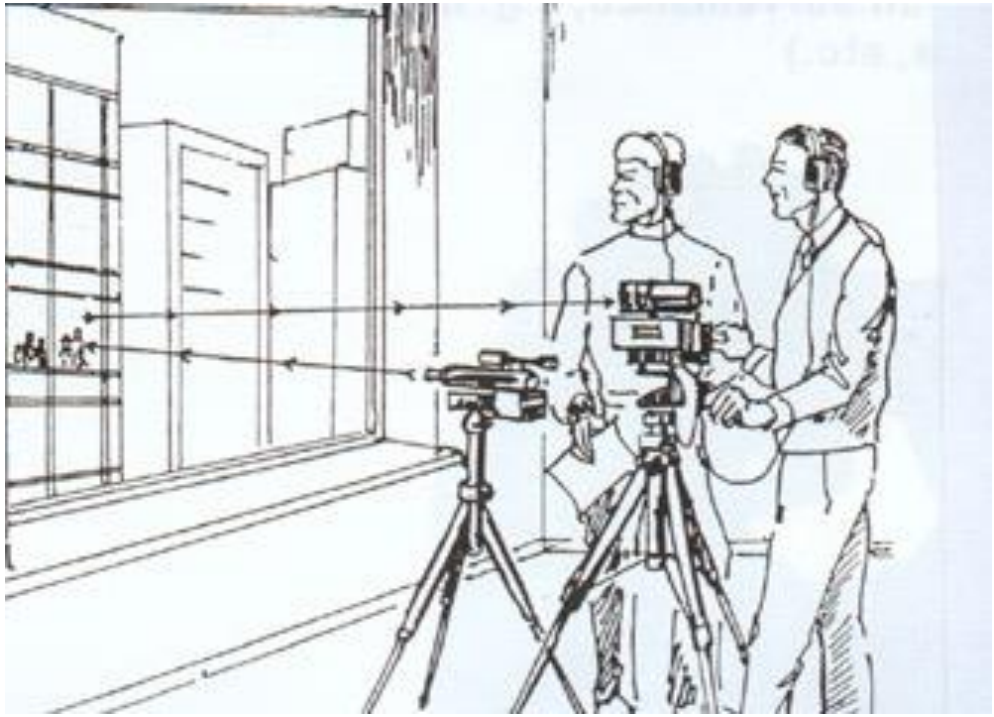
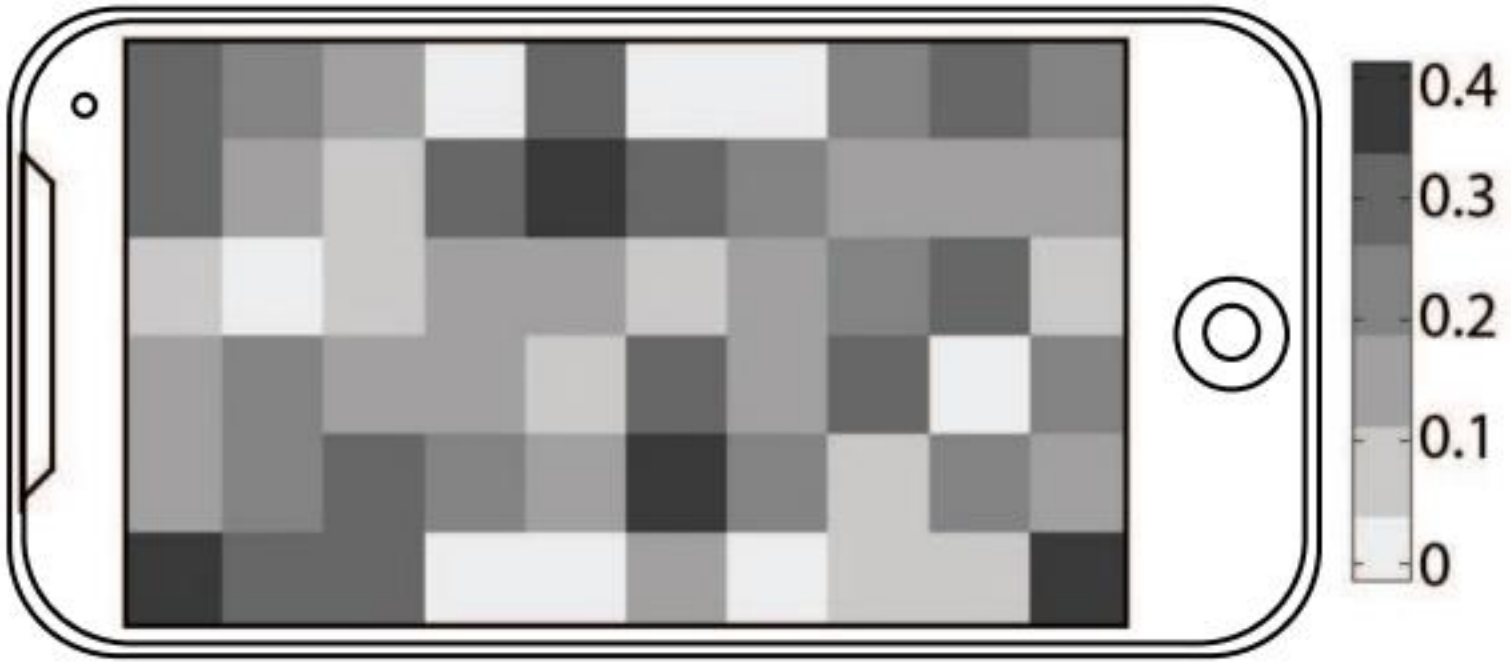


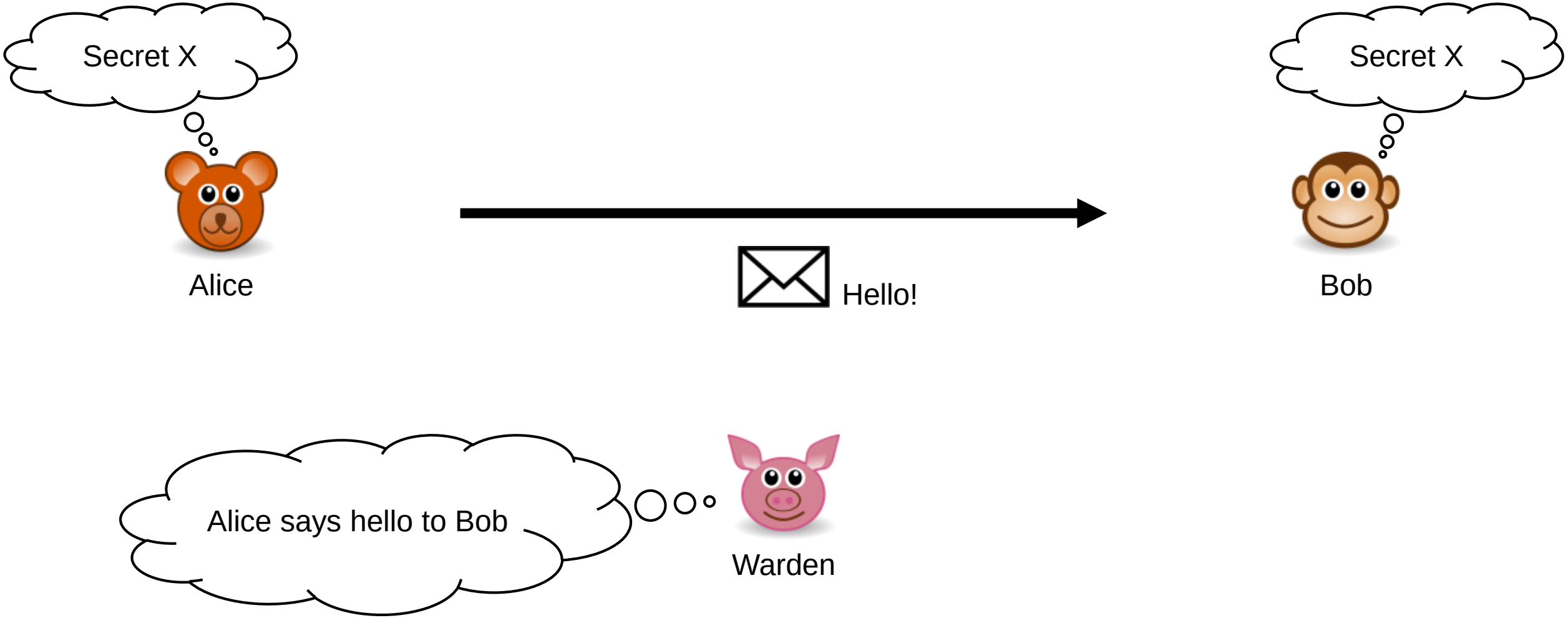
Image from https://i.stack.imgur.com/qoaJh.jpg

Example: Keystroke Inference using Accelerometers



* ACCessory: Password Inference using Accelerometers on Smartphones, ***HotMobile 2012***

Example: Steganography



Example: Steganography

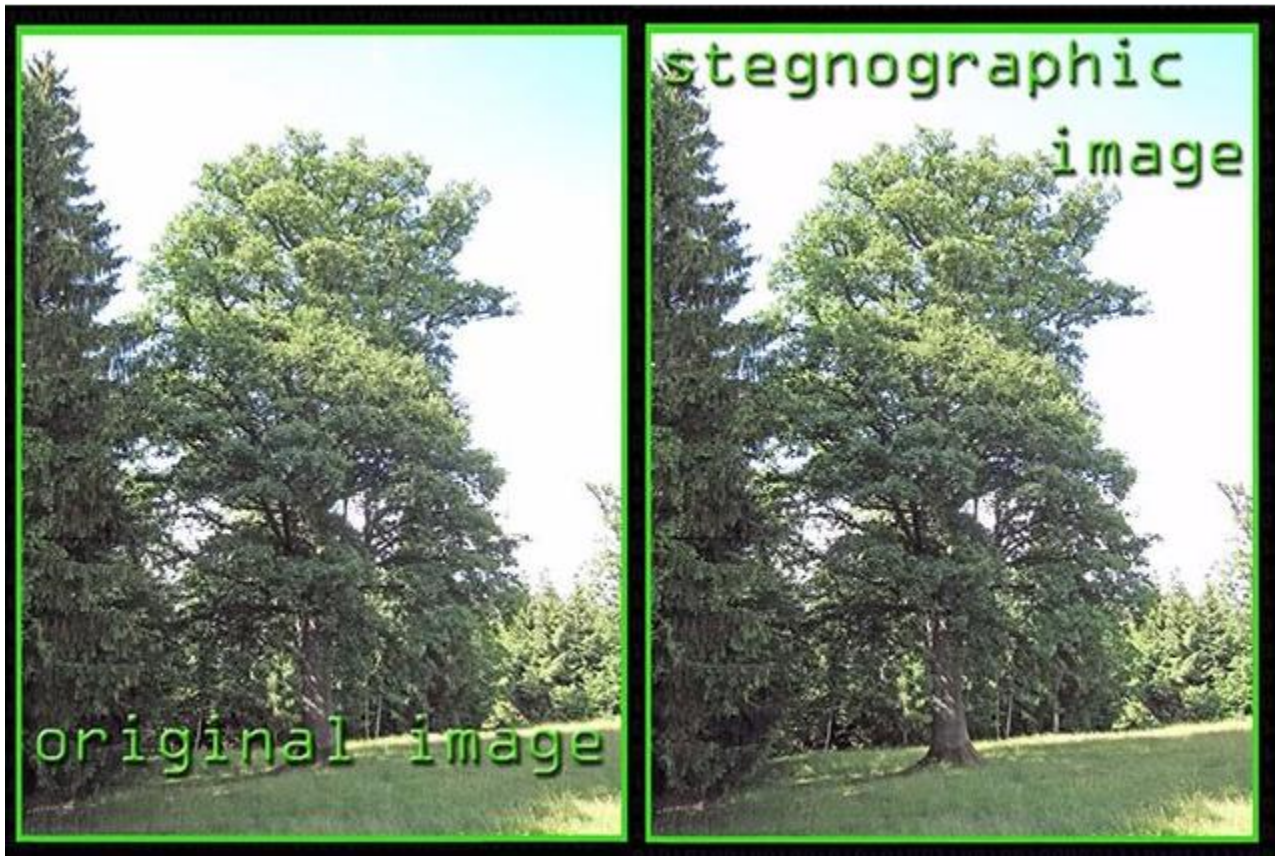


Image from https://petapixel.com/2015/08/07/a-look-at-photo-steganography-the-hiding-of-secrets-inside-digital-images/

Example: Printer Sound

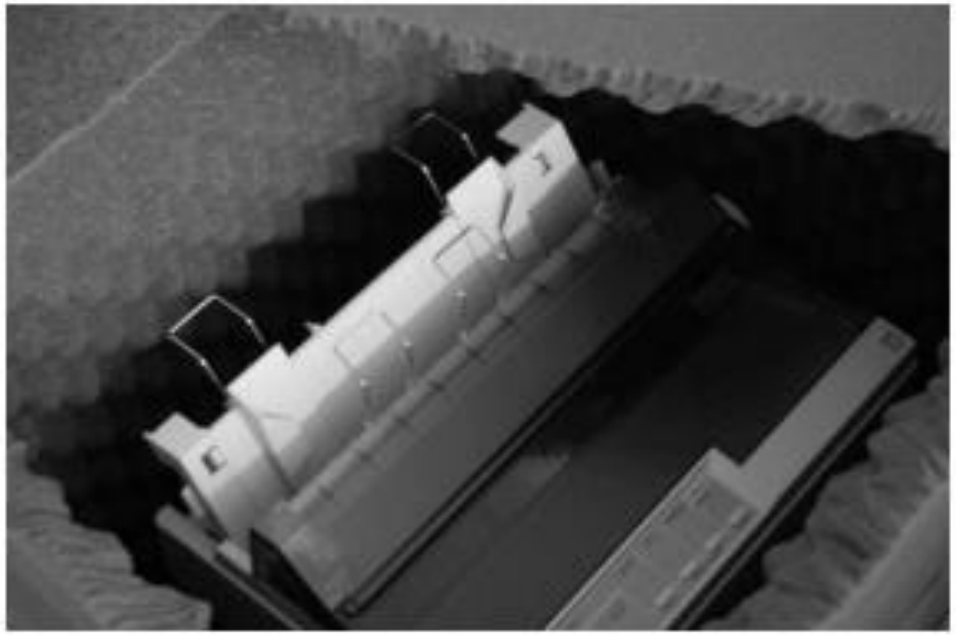


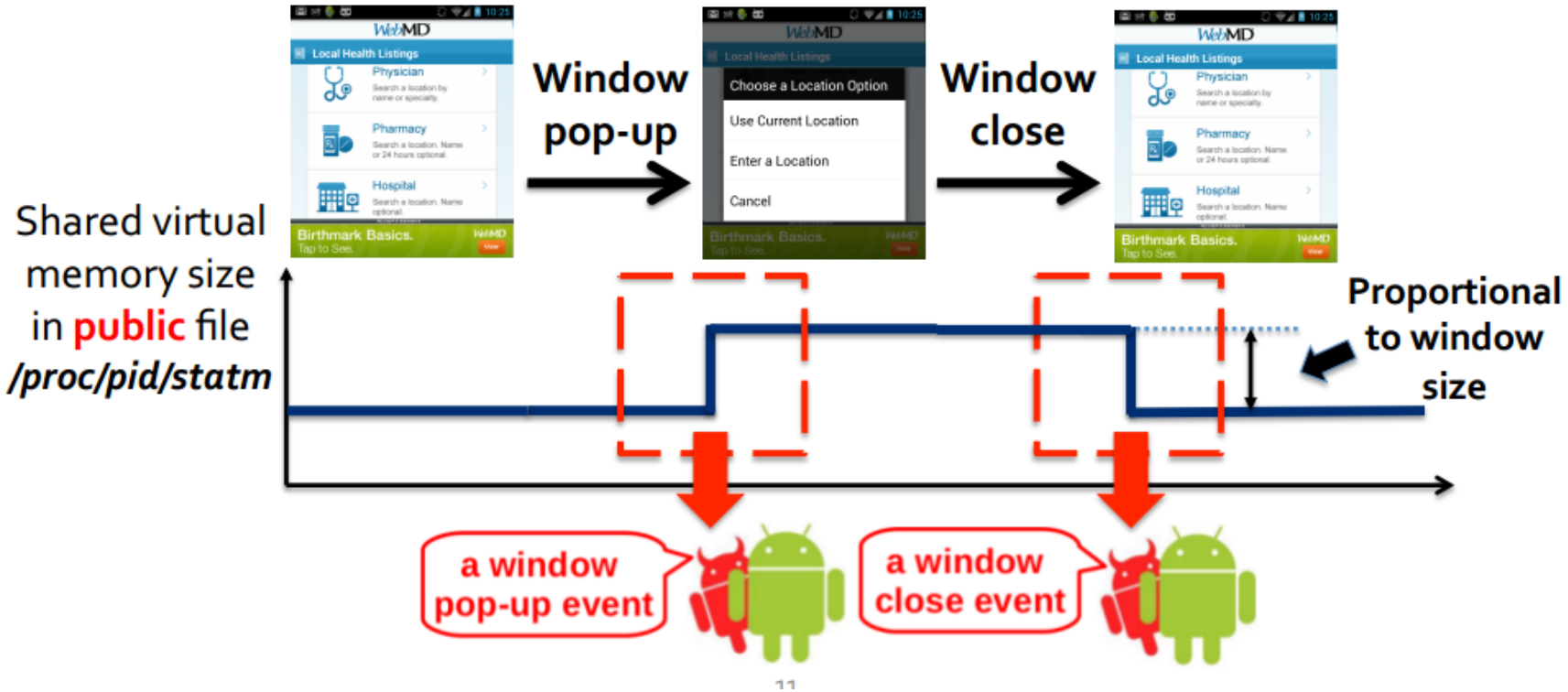
Figure 12: Printer in foam box for shielding evaluation.

Figure 13: The setup of the in-field attack.

* Acoustic Side-Channel Attacks on Printers, ***USENIX Security 2010***

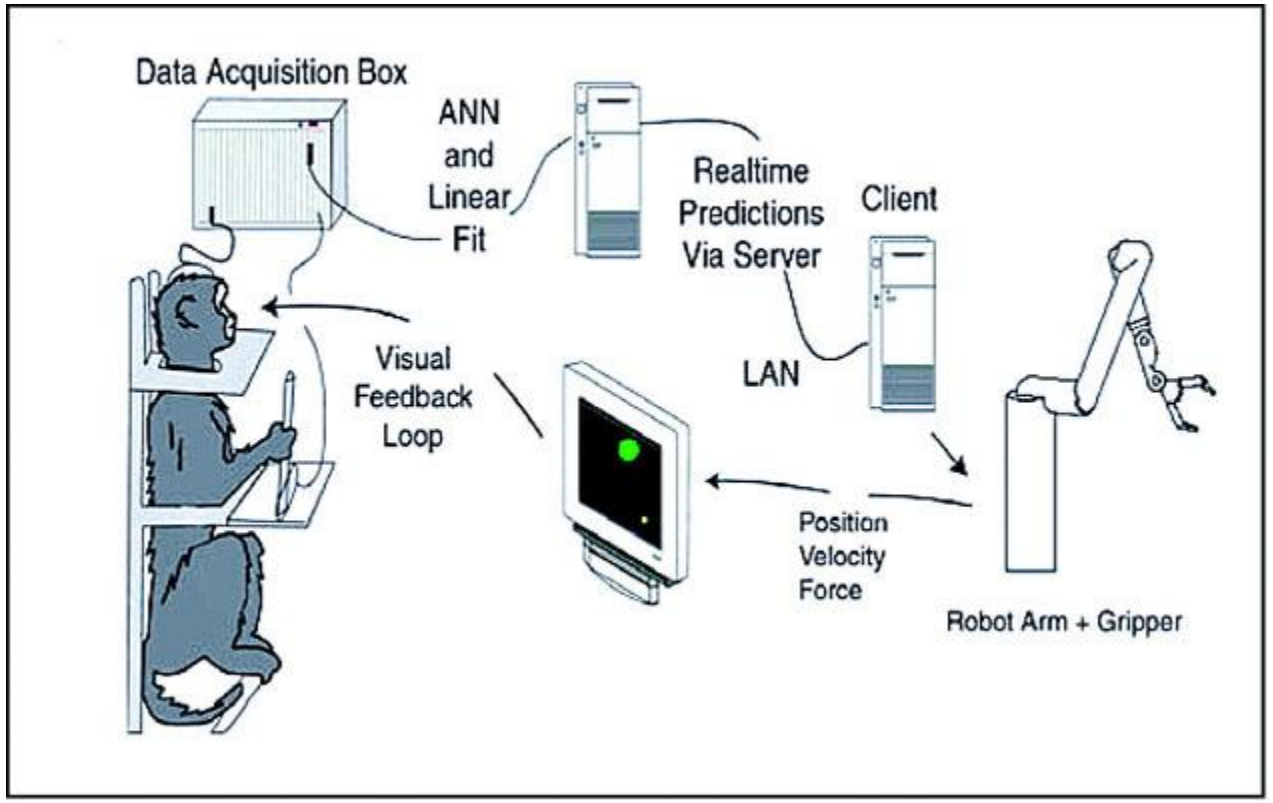
Example: OS Shared Memory

- **Finding**: shared virtual memory size changes are correlated with Android window events



* Taken from Peeking into Your App without Actually Seeing It: UI State Inference and Novel Android Attacks, USENIX Security 2014

Example: Brain Computer Interface



* https://en.wikipedia.org/wiki/Brain%E2%80%93computer_interface

Example: Brain Computer Interface

* On the Feasibility of Side-Channel Attacks with Brain-Computer Interfaces, ***USENIX Security 2012***

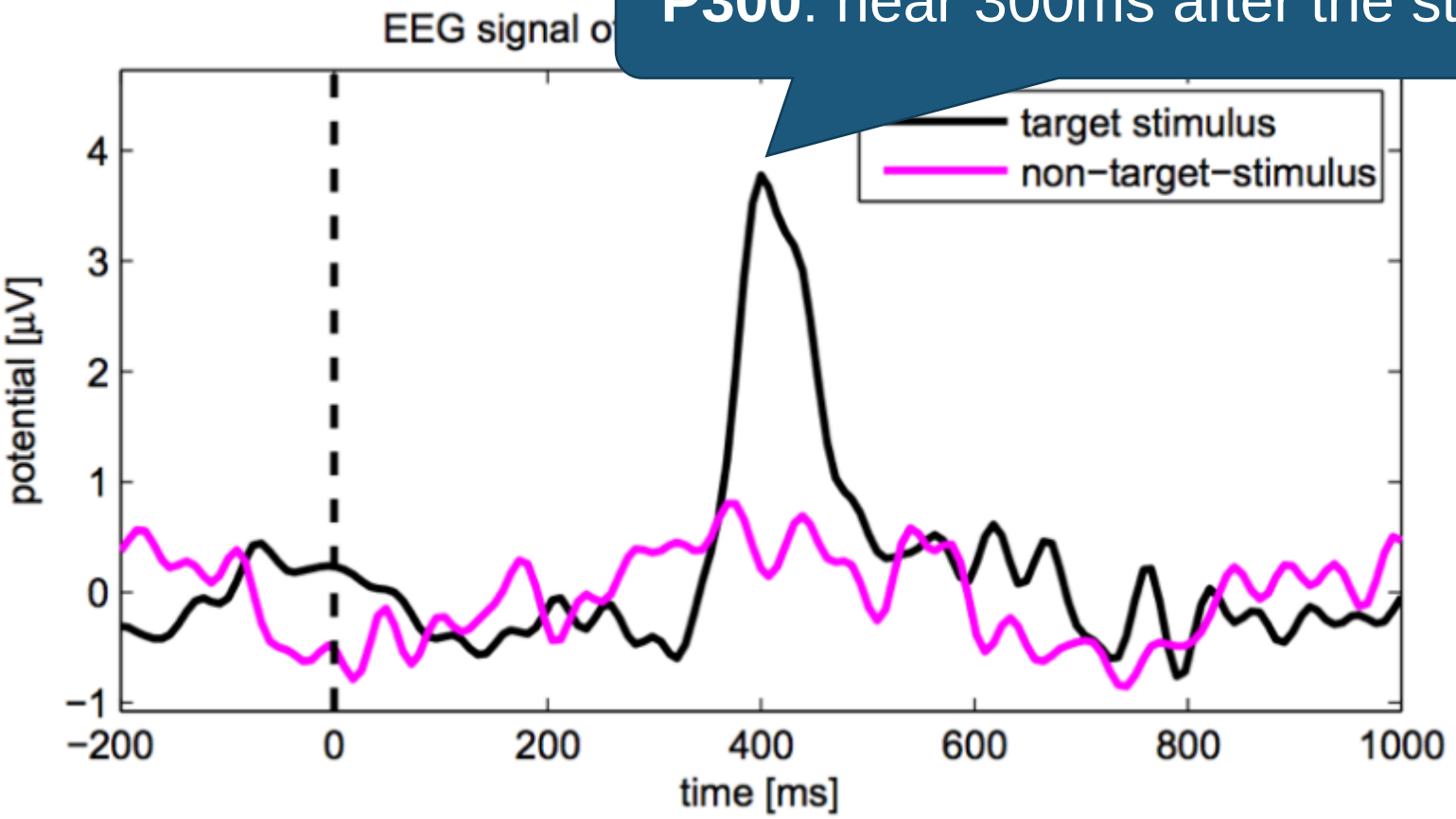
Example: Brain Computer Interface

Can EEG (electroencephalography) applications infer private information about the users by manipulating the visual stimuli presented on screen?

* On the Feasibility of Side-Channel Attacks with Brain-Computer Interfaces, ***USENIX Security 2012***

Event Related Potential

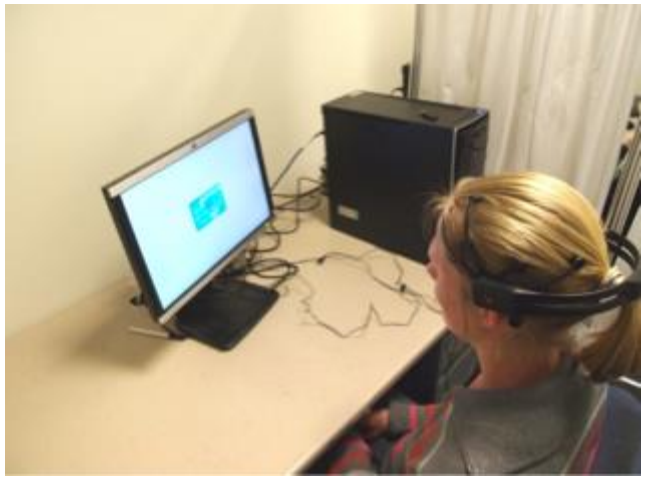
P300: near 300ms after the stimulus



* On the Feasibility of Side-Channel Attacks with Brain-Computer Interfaces, ***USENIX Security 2012***

Attack Model

The attacker can read the EEG signal from the device and can display text, videos, and images on the screen.



* On the Feasibility of Side-Channel Attacks with Brain-Computer Interfaces, ***USENIX Security 2012***

The Attack

- ***Training phase***
  Given a random number x (0-9), ask a user to count the number of occurrence of x from a randomly permuted sequence of numbers from 0 to 9.

- ***Experiment 1***
  Generate a random 4-digit PIN number, and ask a user to memorize it. No special instruction is given (e.g., no need to count the number of occurrence). Randomly permuted sequence of numbers from 0 to 9 were shown to the user.

The Attack

- ***Experiment 2***
  Show the question “what is the name of your bank?” for 2

From the paper: we show that the entropy of the private information is decreased on the average by approximately 15 % - 40 % compared to random guessing attacks.

- ***Experiment 4****:* Face recognition
- ***Experiment 5****:* Geographic location

* On the Feasibility of Side-Channel Attacks with Brain-Computer Interfaces, ***USENIX Security 2012***

Example: CPU Pipeline

8.2.3.4 Loads May Be Reordered with Earlier Stores to Different Locations

The Intel-64 memory-ordering model allows a load to be reordered with an earlier store to a different location. However, loads are not reordered with stores to the same location.

The fact that a load may be reordered with an earlier store to a different location is illustrated by the following example:

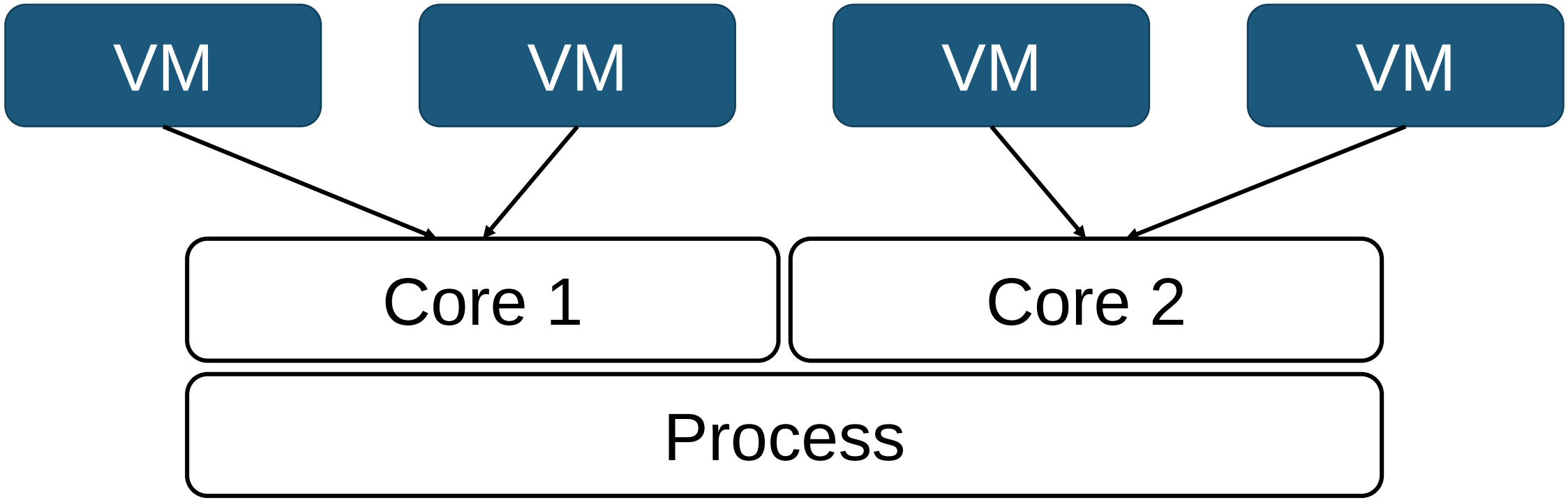
Example 8-3. Loads May be Reordered with Older Stores

Processor 0	Processor 1
mov [_x], 1 mov r1, [_y]	mov [_y], 1 mov r2, [_x]
Initially x = y = 0 r1 = 0 and r2 = 0 is allowed	

From Intel manual Vol. 3A 8-9

Out of Order Execution

Maximizing the use of CPU pipeline's cycles



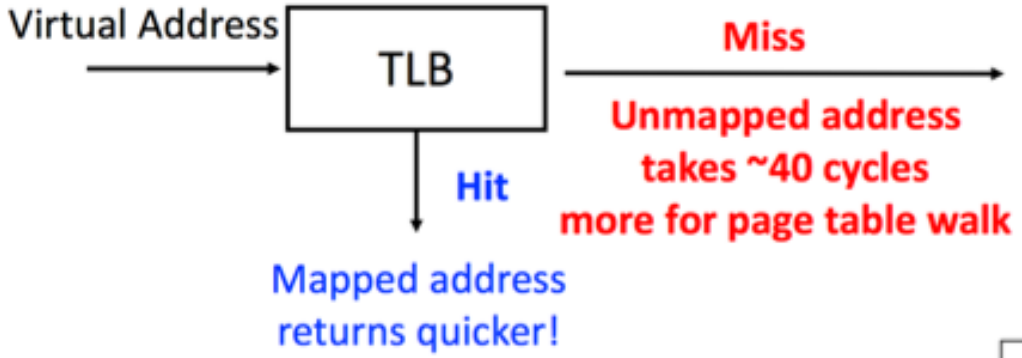
CPU Pipeline Covert Channel

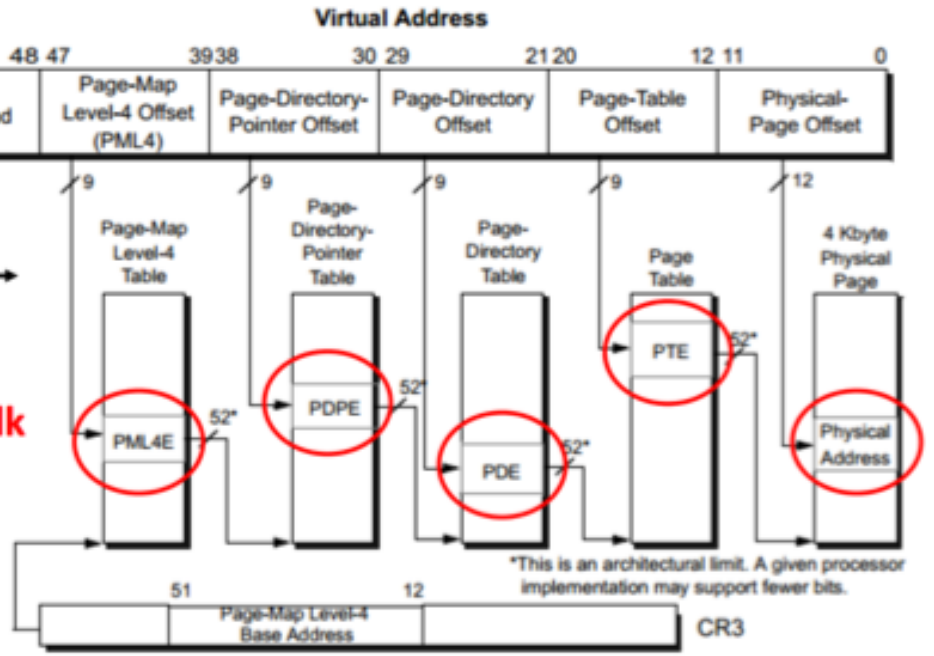
- Transmitter
  - Use `mfence` instruction to prevent reordering
  - Given a time frame, turn on/off out-of-order executions

- Receiver
  - Count the number of out-of-order executions for each time frame (compute out-of-order-execution frequency)
  - Know whether oooe is on/off (binary information)

Exploiting out-of-order execution, ***Blackhat USA 2015***
Out-of-Order Execution as a Cross-VM Side-Channel and Other Applications, ***ROOTS 2017***

Example: TLB Timing Channel





* Image taken from Yeongjin Jang's Blackhat USA 2016 talk

Intel TSX

- Transactional Synchronization eXtensions (TSX)
- Hardware-level memory transaction
  - XBEGIN/XEND instructions
  - Speed up multi-threaded applications

- When a XBEGIN/XEND block tries to access kernel memory, no page fault is raised, it just aborts the transaction
  - Execution ***never leaves user mode***
  - Less CPU clocks used, and present better precision on timing attack

TSX-based Timing Attack

```
uint64_t time_begin, time_diff;
int status = 0;
int *p = (int*)0xffffffff80000000; // kernel addresss
time_begin = __rdtscp();
if((status = _xbegin()) == _XBEGIN_STARTED) {
    // TSX transaction
    *p; // read access
    // or,
    ((int(*)())p)(); // exec access
}
else {
    // abort handler
    time_diff = __rdtscp() - time_begin;
}
```

1. Timestamp at the beginning

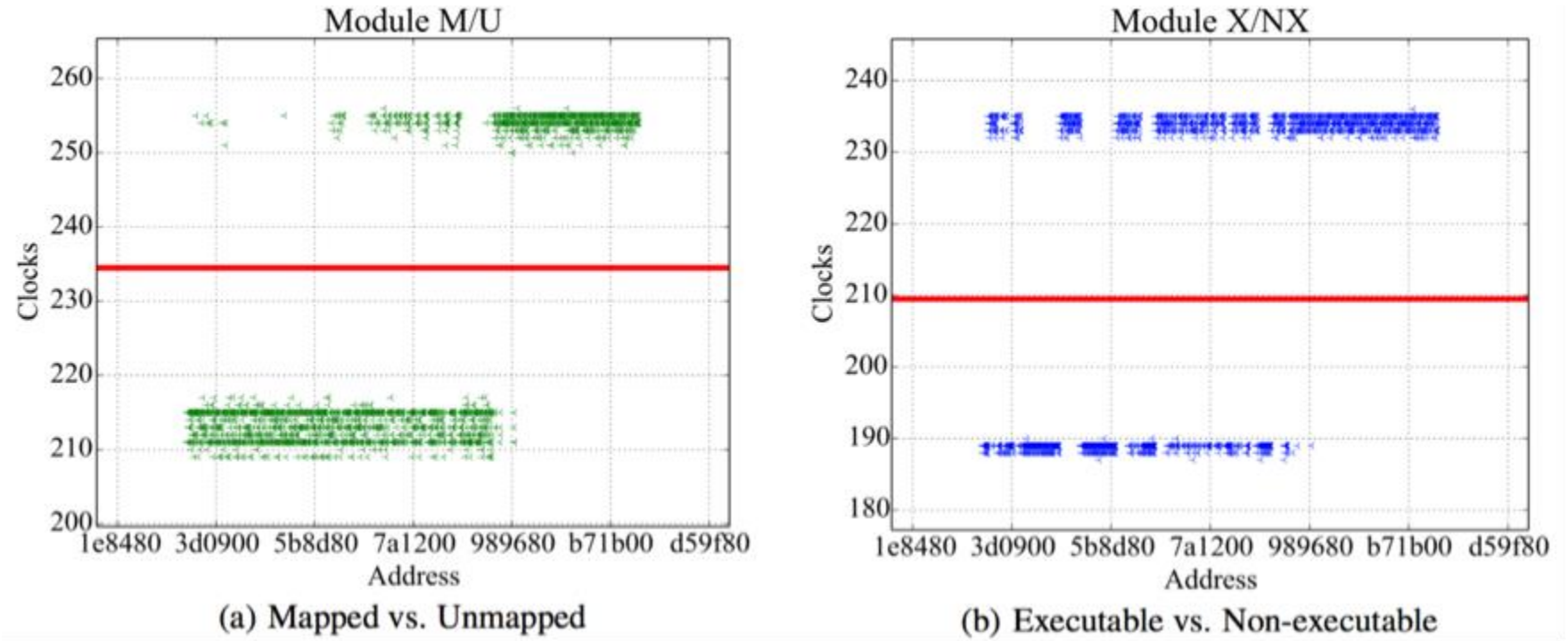
2. Access kernel memory within the TSX region (always aborts)

No OS handling path is involved

3. Measure timing at abort handler

* Image taken from Yeongjin Jang's Blackhat USA 2016 talk

Clear Timing Channel



* Image taken from Yeongjin Jang's Blackhat USA 2016 talk

Example: Meltdown and Spectre

Image from https://meltdownattack.com/

Speculative Execution

```
// case 1
if (input < len1) {
  value = data[input]; // Speculatively executed and
}                      // even cached!

// case 2
if (input < len1) {
  value = data[input];
  addr = (value & 1) * 0x100 + 0x200;
  if (addr < len2) {
    bit = data[addr]; // Is this cached or not?
```

Conclusion

- Covert channel vs. side channel
- Always watch out the shared resources

Question?