Advanced topics in bird-related security

Secure Systems Engineering Spring 2024

🛡️ EE G7701

April 9, 2024

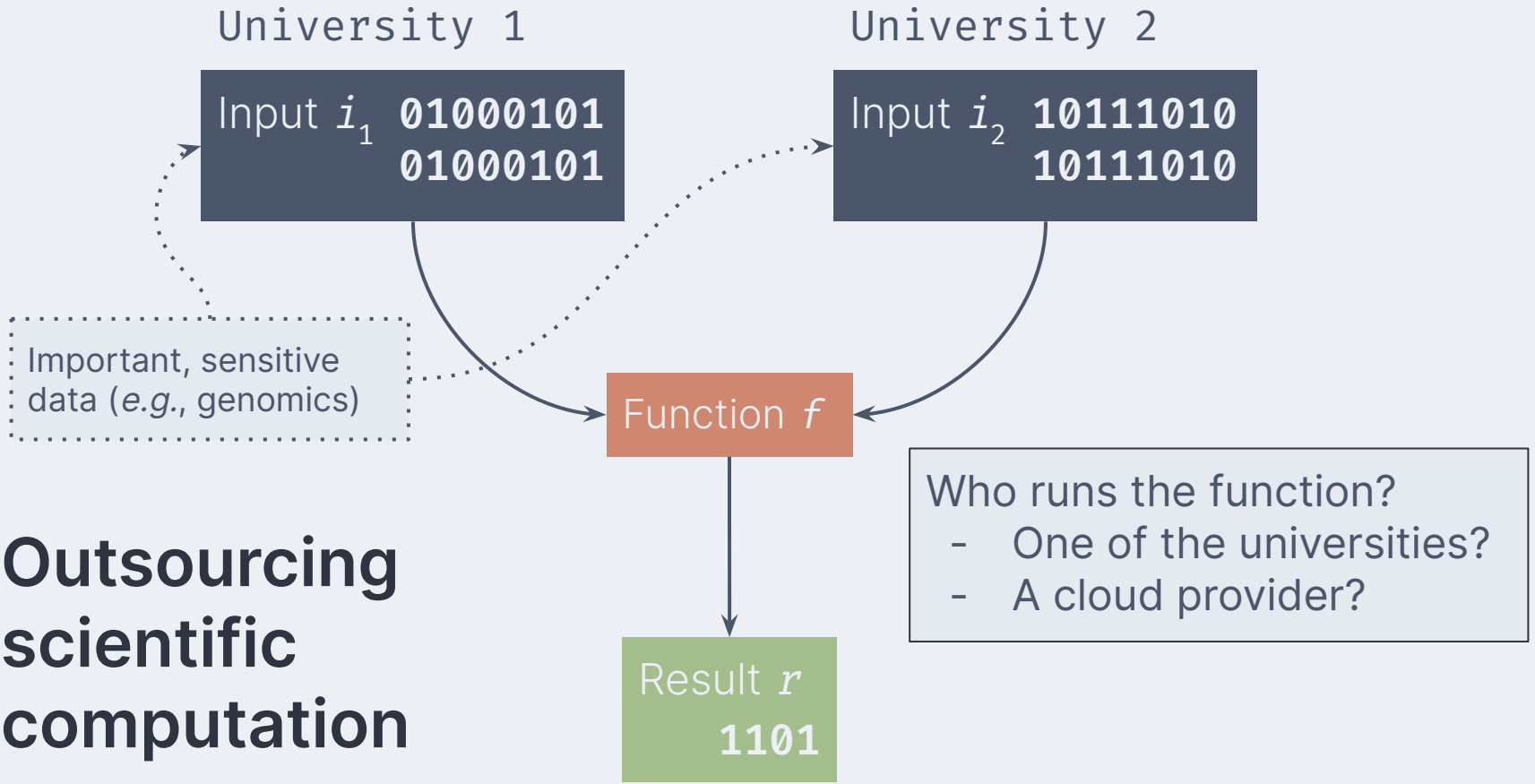
Tushar Jois

Recap

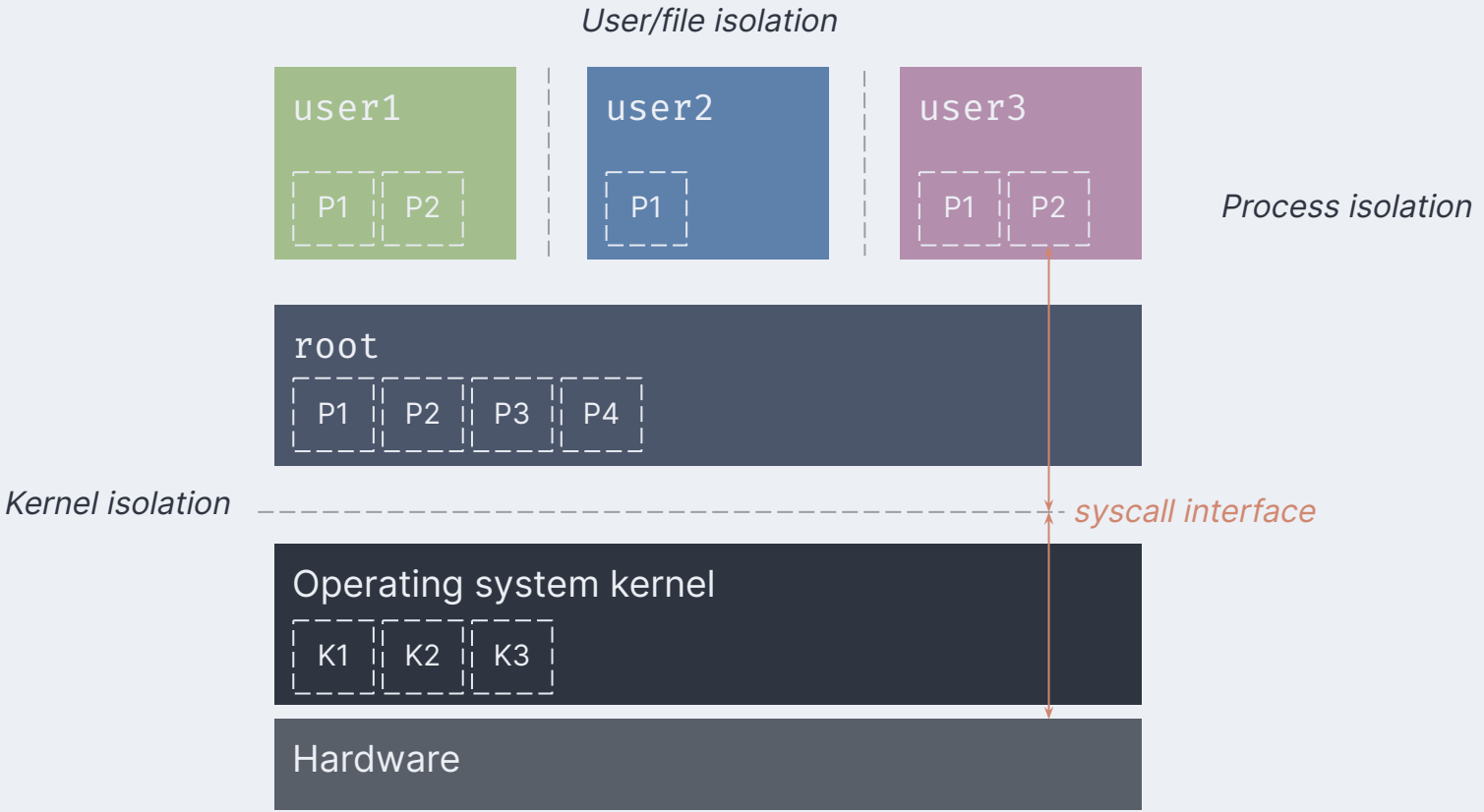
- Onion routing in Tor builds circuits to provide censorship resistance
- The Signal protocol provides forward secrecy through its primitives
- To use privacy technology in practice is to navigate political questions

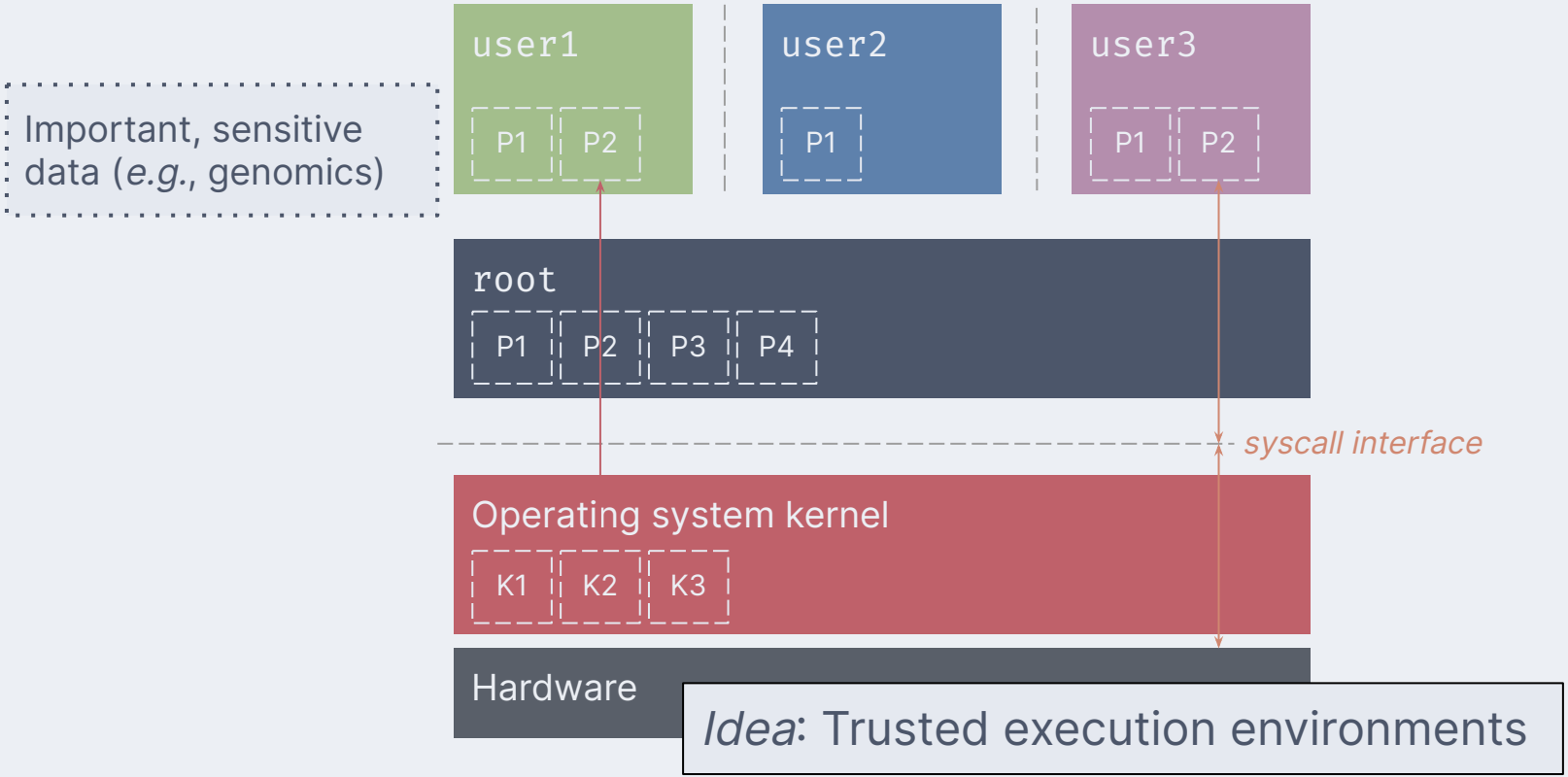
Lesson objectives

- Understand the problems of outsourcing scientific computation securely
- Apply data-oblivious computation to address microarchitectural side-channel attacks
- Be prepared for the second midterm exam in the course



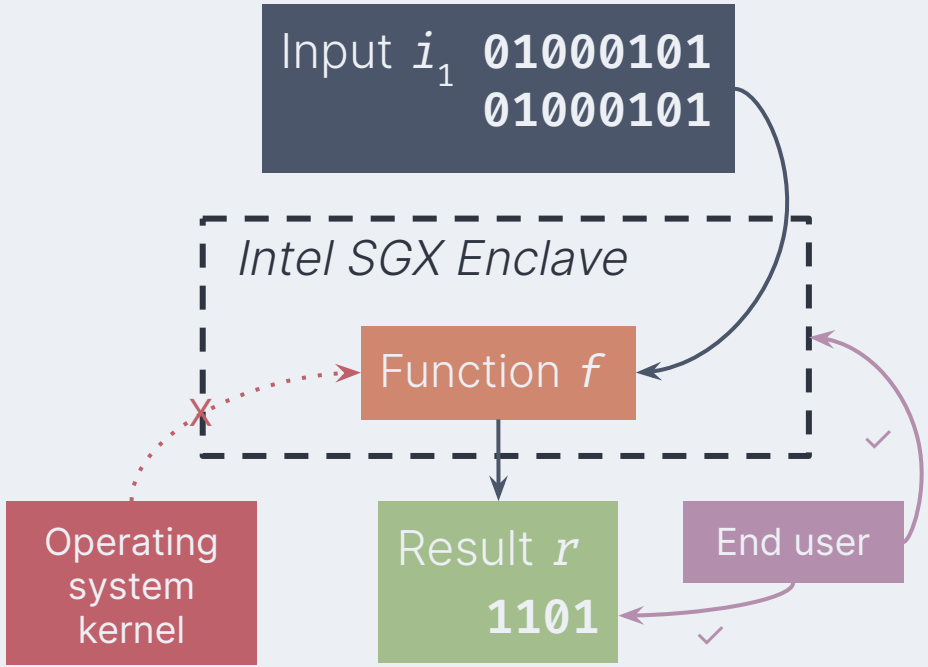
Outsourcing scientific computation

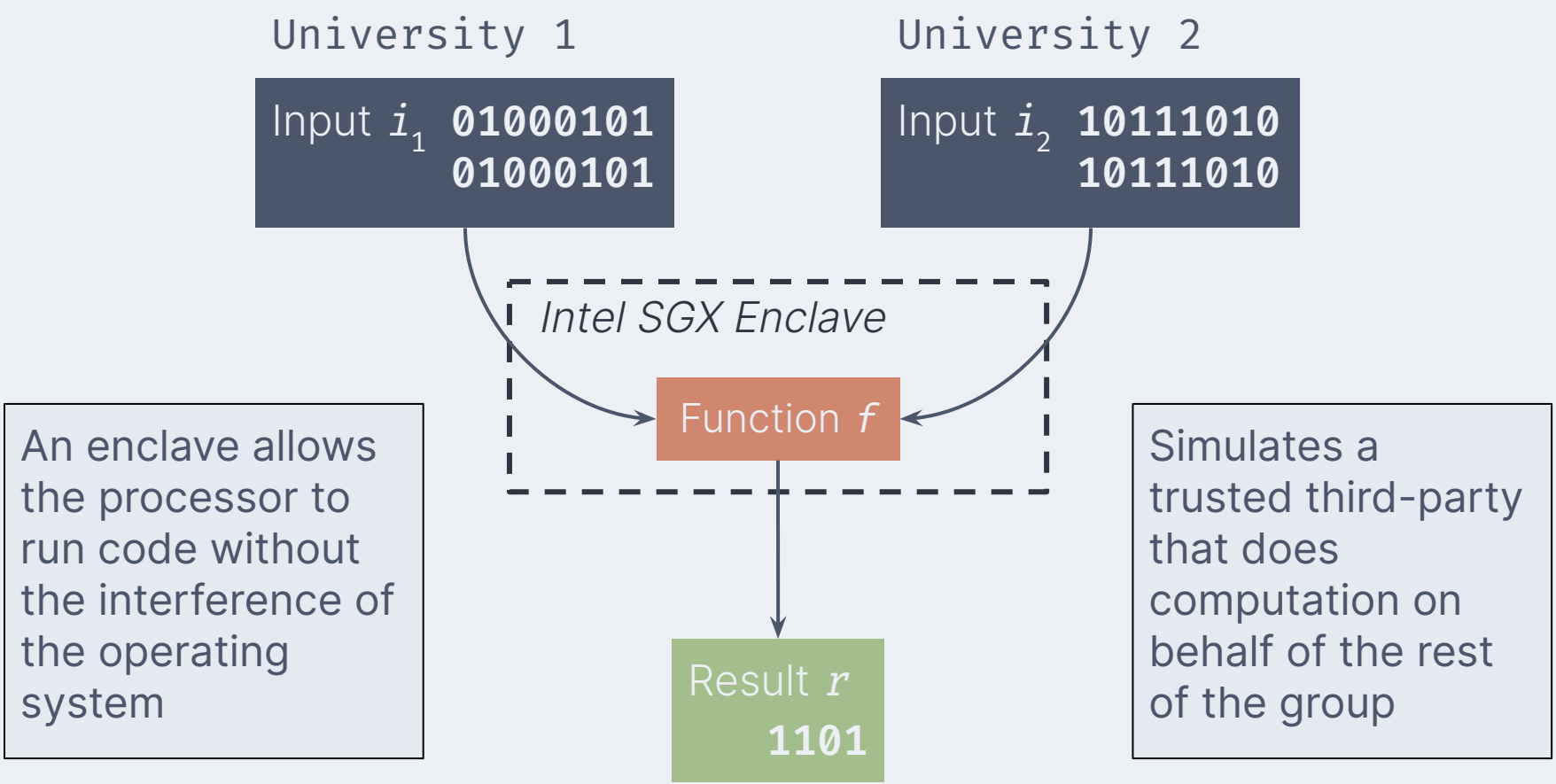

User/file isolation
user1
P1
P2
user2
P1
user3
P1
P2
Process isolation
root
P1
P2
P3
P4
Kernel isolation
syscall interface
Operating system kernel
K1
K2
K3
Hardware

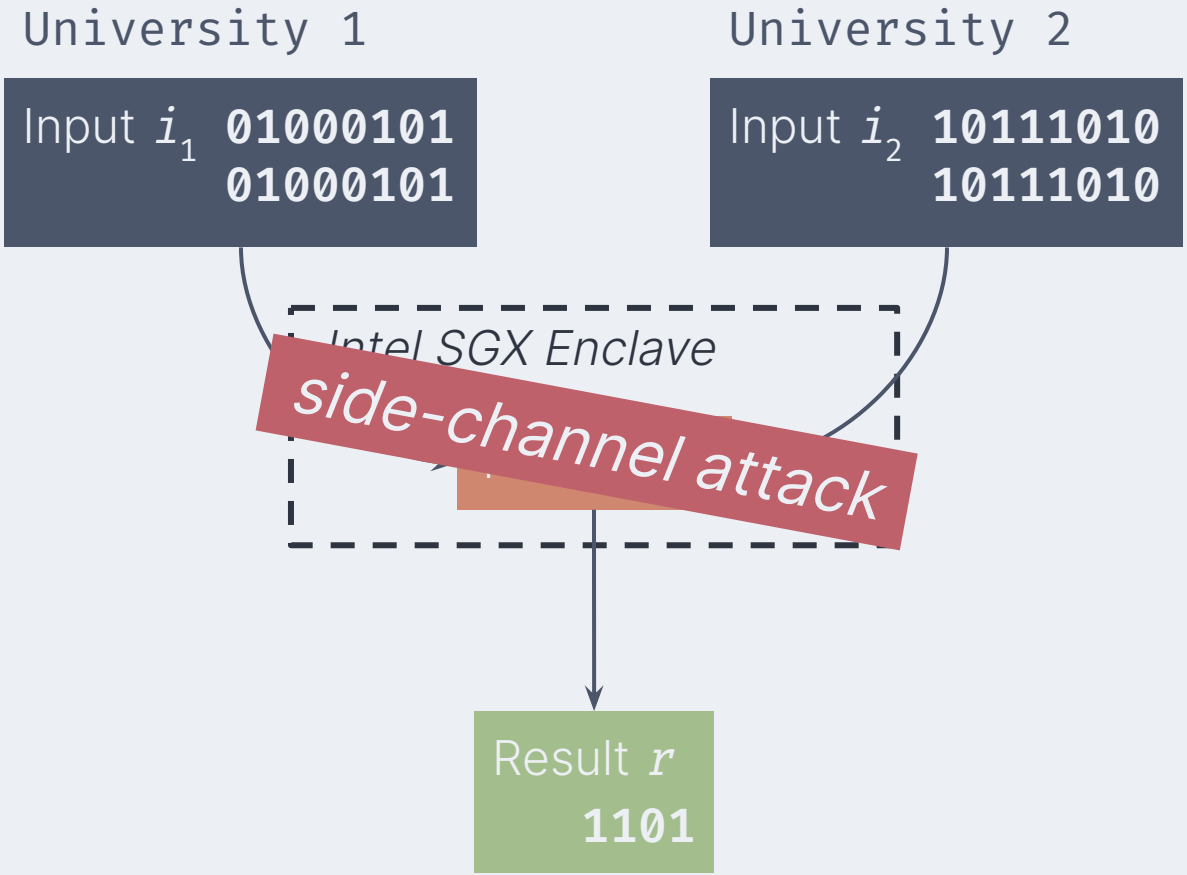

Important, sensitive data (*e.g.*, genomics)
user1
P1
P2
user2
P1
user3
P1
P2
root
P1
P2
P3
P4
syscall interface
Operating system kernel
K1
K2
K3
Hardware
Idea: Trusted execution environments

Trusted execution environment

- A separate “part” of the processor
  - Untrusted applications run normally
  - Trusted applications run in an *enclave*
- When a processor computes on an enclave, it encrypts its memory
  - The operating system can’t access the memory/data unless it has the processor’s key
- Enclaves can use *remote attestation*
  - The processor signs what it’s executing
  - You can be sure nothing has changed
- Feature requires hardware support
  - Combines cryptography and systems security together

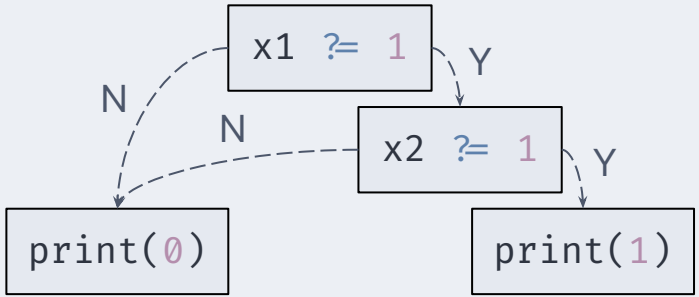



University 1
Input i_1 01000101 01000101
University 2
Input i_2 10111010 10111010
Intel SGX Enclave
Function f
Result r 1101
An enclave allows the processor to run code without the interference of the operating system
Simulates a trusted third-party that does computation on behalf of the rest of the group


University 1
Input i_1 01000101
01000101
University 2
Input i_2 10111010
10111010
Intel SGX Enclave
side-channel attack
Result r
1101

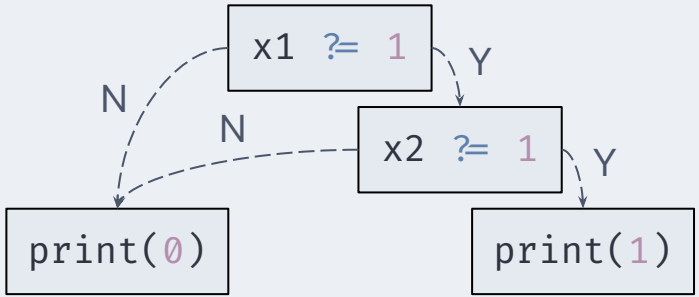
```python
# Side-channel Vulnerable
# Assume x1, x2 are private

if x1 and x2:
  print(1)
else:
  print(0)
```

Execution Trace



```
# Side-channel Vulnerable
# Assume x1, x2 are private

if x1 and x2:
  print(1)
else:
  print(0)
```

Execution Trace



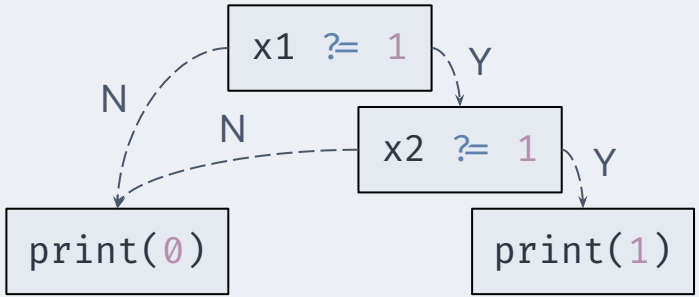
microarchitectural side-channel

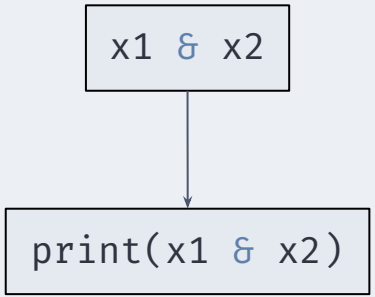
An attacker can infer the inputs of an operation by looking at the state of the processor during its execution.

```
# Side-channel Vulnerable
# Assume x1, x2 are private

if x1 and x2:
  print(1)
else:
  print(0)
```

```
# Fixing the vulnerability

print(x1 & x2)
```

Execution Trace





da·ta-o·bliv·i·ous com·pu·ta·tion (*n.*)

a program execution with the same observable characteristics regardless of the inputs provided

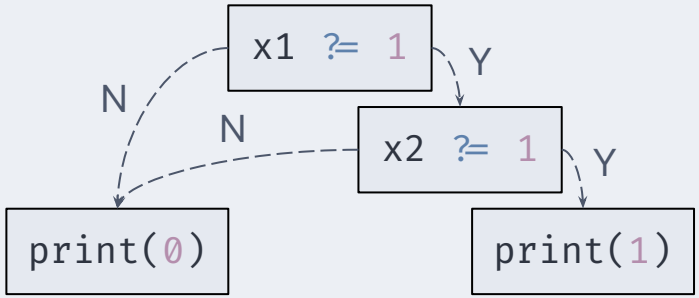
see also constant-time programming
antonym data-dependent computation

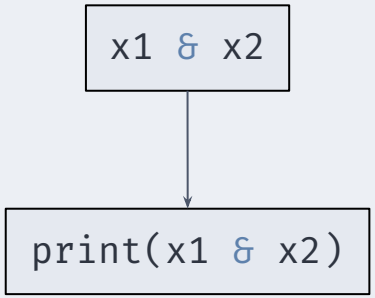
```
# Side-channel Vulnerable
# Assume x1, x2 are private

if x1 and x2:
  print(1)
else:
  print(0)
```

```
# Fixed (under assumptions)


print(x1 & x2)
```

Let's try it!

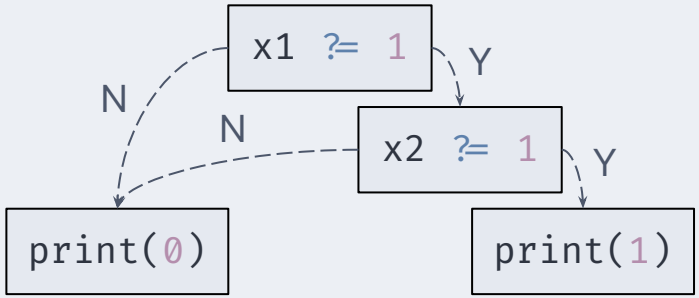
Execution Trace





```python
# Side-channel Vulnerable
# Assume x1, x2 are private

if x1 and x2:
  print(1)
else:
  print(0)
```

Execution Trace



For each function on the right:

- Draw the execution trace
- Determine if the function is data-oblivious with respect to x using the execution trace

```python
def func1(x):
    y = 4
    if x == 6:
        y = 6
    return y
```

```python
def func2(x):
    y = 0
    if x == 6:
        y = 6
    else:
        y = 4
    return y
```

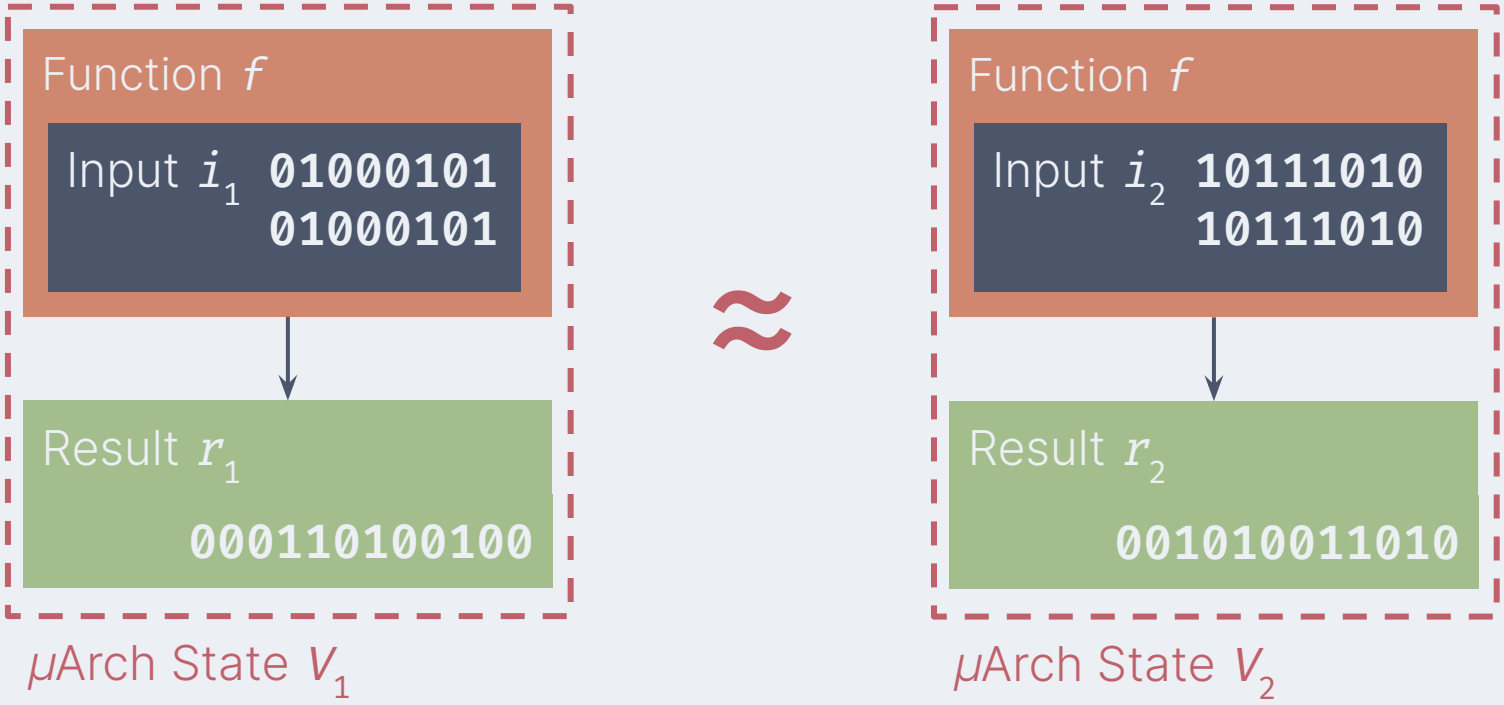
```python
def func3(x):
    y = 1
    if y == 4:
        y = 7
    return y
```

Key insight:

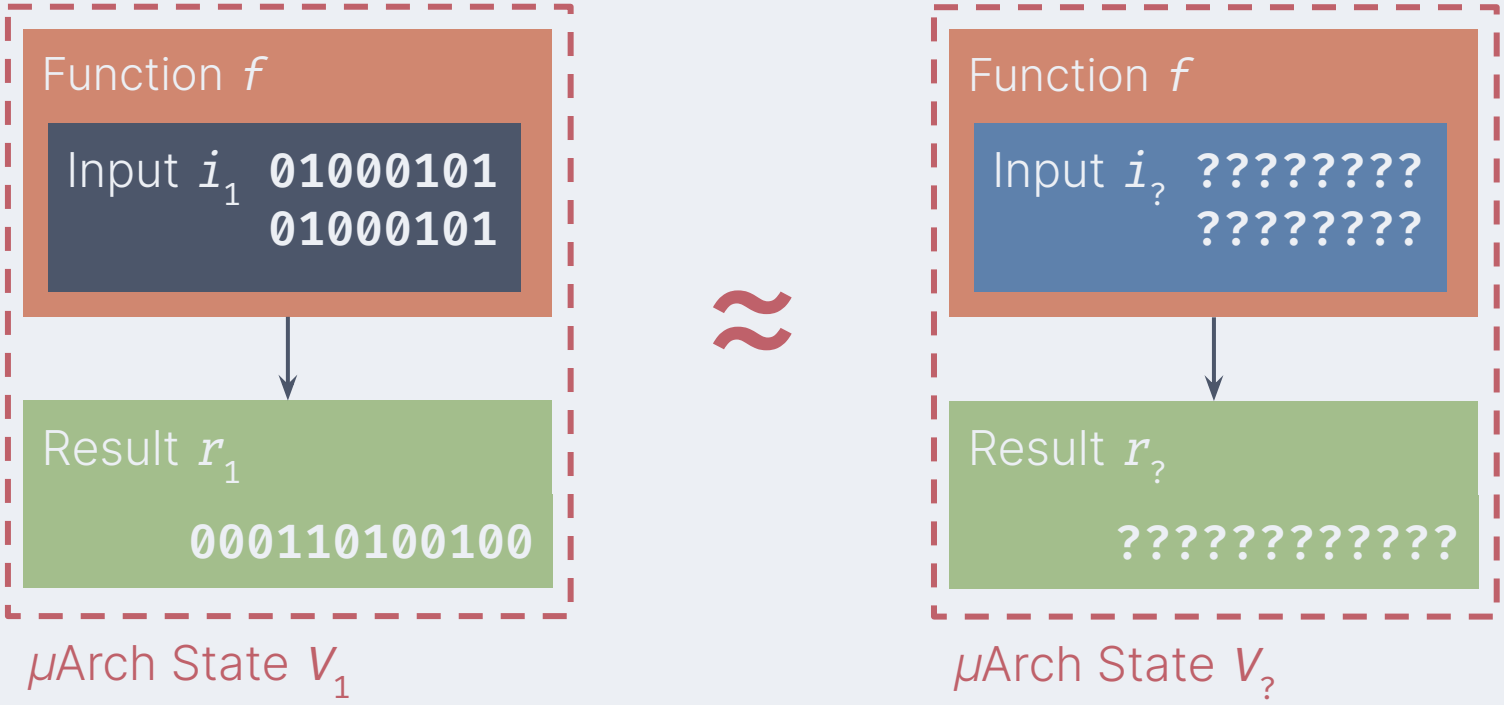
data-oblivious → data-unnecessary

Assuming a **data-oblivious** function

Assuming a **data-oblivious** function



Assuming a **data-oblivious** function



DOVE: A Data-Oblivious Virtual Environment

Hyun Bin Lee, Tushar M. Jois, Christopher W. Fletcher, Carl A. Gunter

Network and Distributed Security Symposium 2021 (NDSS '21)

Function f

Input i_1 **01000101 01000101**

Result r_1 **000110100100**

??????
??????

Data Oblivious Transcript

Input i_1 **01000101 01000101**

Replay Execution Trace on a Dataset

Result r_1 **000110100100**

Separate the execution of a script from the actual operations on sensitive data

Exam 2

Desirable properties of voting systems

Voter feels that:

- Vote was counted
- Vote was private
- Nobody else can vote more than once
- Nobody can alter others' votes

People believe that the machine works correctly and that its behavior cannot be modified

These have to do with perception.

It is also important that these perceptions are true.

"The purpose of an election is to convince the supporters of the losing candidate that they lost"

J. Alex Halderman, University of Michigan

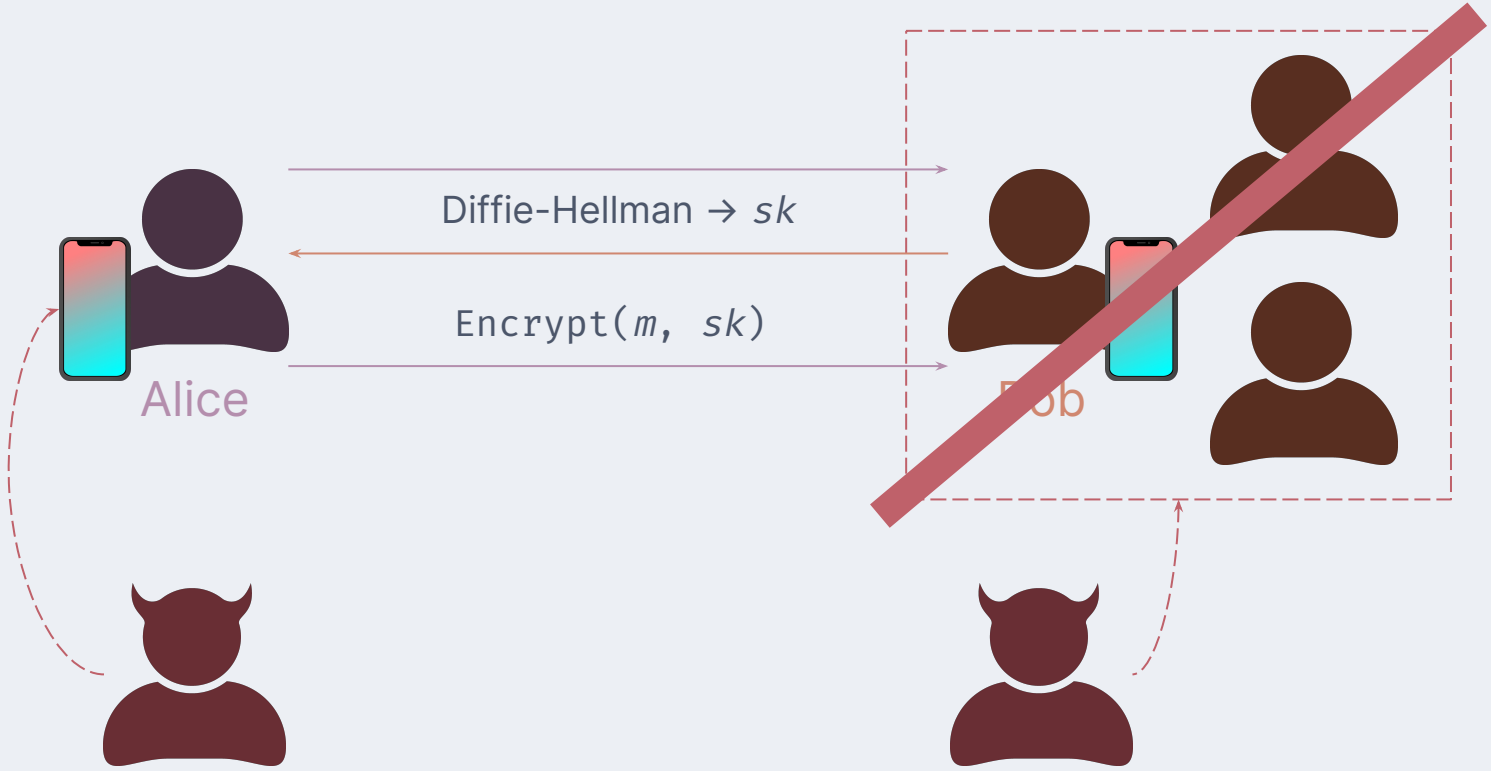
```
// LCG - Linear Conguential Generator
// used to generate ballot serial numbers
// A psuedo-random-sequence generator
// (per Applied Cryptography,
// by Bruce Schneier, Wiley, 1996)
```

`BallotResults.cpp`

Diebold Election Systems

“Unfortunately, linear congruential generators **cannot be used for cryptography**”

Bruce Scheiner
Applied Cryptography (Wiley, 1996)
Page 369


Diffie-Hellman → sk
Encrypt(m, sk)
Alice
Bob


TRAFO. ST. 10

A backdoored PRNG

s_k — Internal PRNG states

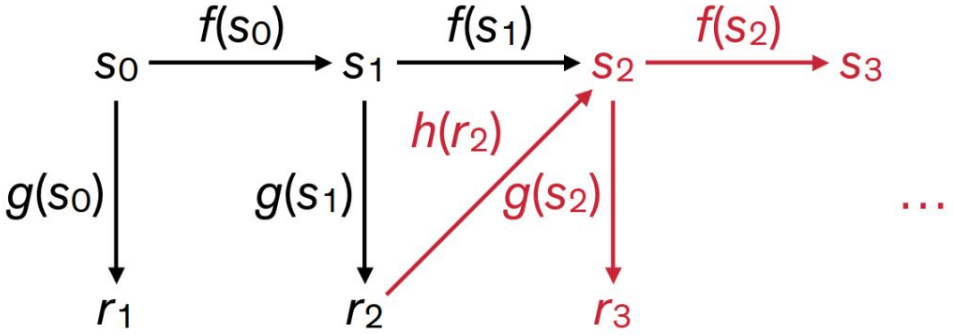
r_k — Outputs

$f(\bullet)$ — State update function

$g(\bullet)$ — Output function

$h(\bullet)$ — Backdoor function

■ — Attacker computation



ScreenOS 6.2 PRNG

```
char output[32];     // PRNG output buffer
int  index;          // Index into output
char seed[8];        // X9.31 seed
char key[24];        // X9.31 key
char block[8];       // X9.31 output block
int  reseed_counter;

void x9_31_reseed(void)
  reseed_counter = 0;
  if (dualec_generate(output, 32) != 32)
    error("[...]PRNG failure[...]", 11);
  memcpy(seed, output, 8);
  index = 8;
  memcpy(key, &output[index], 24);
  index = 32;
}
```

32 bytes from Dual EC stored in output

index set to 32

```
void prng_generate(void) {
  int time[2] = { 0, g
  index = 0;
  ++reseed_counter;
  if (!one_stage_rng())
    x9_31_reseed();
  for (; index < 32; index += 8) {
    // FIPS checks removed for clarity
    x9_31_gen(time, seed, key, block);
    // FIPS checks removed for clarity
    memcpy(&output[index], block, 8);
  }
}
```

index set to 0

Always returns false*; reseed on every call

Loop never executes!

output still contains 32 bytes from Dual EC

★ Can be disabled

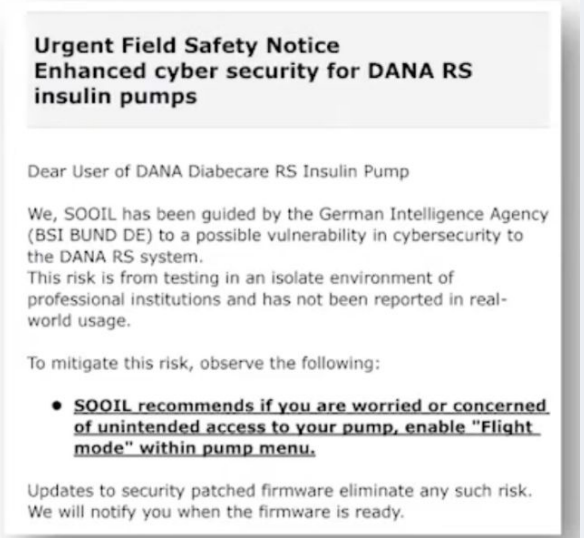
Threat Model

- Patient harm and non-harm risks
  - Pump delivers too much, not enough insulin
  - Pump leaks logs to unauthorized users

- Microsoft STRIDE
  - **S**poofing: Impersonate system
    - Attacker masquerades as smartphone
  - **T**ampering: Modify data or code
    - Attacker MITMs pump-to-smartphone communication
  - **R**epudiation: Claiming to not have performed an action
    - Pump audit log cannot distinguish unauthorized commands

Remediation, Temporary Measures & Disclosure

- Remediations are Complex:
  - Most vulnerabilities are design defects
  - The manufacturer prepared and rolled out a firmware update for the insulin pump in 04/2020.
- Operations is key:
  - Disable the insulin pump's BLE functionality → airplane mode→ Preserve the pump's therapeutic purpose
  - The device implements safety features
- Disclosure managed by the ManiMed project team
  - A publication of the vulnerabilities does not pose serious risks or harm to patients as short-term measures or workarounds exist that preserve the pump's therapeutic purpose



Urgent Field Safety Notice
Enhanced cyber security for DANA RS insulin pumps

Dear User of DANA Diabecare RS Insulin Pump

We, SOOIL has been guided by the German Intelligence Agency (BSI BUND DE) to a possible vulnerability in cybersecurity to the DANA RS system.
This risk is from testing in an isolate environment of professional institutions and has not been reported in real-world usage.

To mitigate this risk, observe the following:

- **SOOIL recommends if you are worried or concerned of unintended access to your pump, enable "Flight mode" within pump menu.**

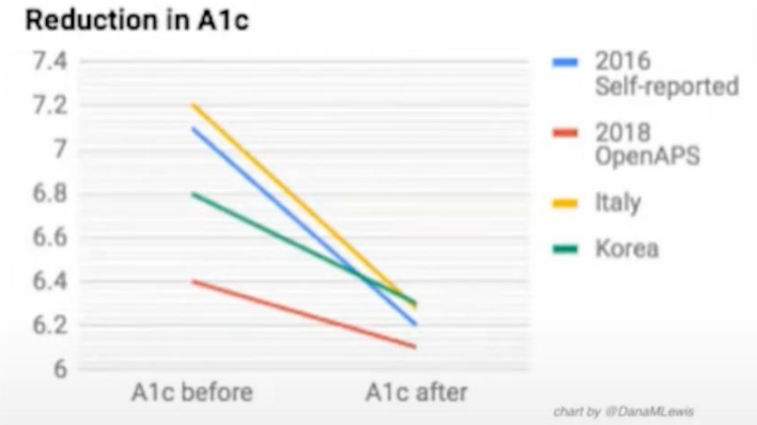
Updates to security patched firmware eliminate any such risk. We will notify you when the firmware is ready.

[https://advancedtherapeutics.org.uk/wp-content/uploads/2020/03/User-FSN-DANA-RS-Cybersecurity-002-6-March-2020.pdf, accessed July 28,2020]

DIY Diabetics Community

- Julian, Dina's disclosure stirred up mixed feelings from the community
  - DANA RS remediations broke AndroidAPS compatibility

- Public service announcement:
  - We support the DIY Diabetics community, their hard work, and dedication
    - 11,896+ people using DIY Closed-loop systems as of May 11th, 2020[1]
    - 24,000,000+ loop hours[1]



1. https://openaps.org/outcomes/

The importance of secure messaging

- Facebook Messenger, Instagram are not “end-to-end”
  - Facebook reads the messages, delivers ads about them
  - Governments can subpoena Facebook for your messages, reconstruct your digital life
- “Surveillance capitalism”
  - The person is the product
  - “Free” services provided by Big Tech powered by the selling of your data
- Data sharing agreements
  - Seen ads for things you’ve talked about on Amazon?

“But I have nothing to hide!”

- Solidarity with those who do
  - Snowden/whistleblowers, but also “The Feeling of Being Watched” subjects
- You might not realize how much data is out there
  - “We kill people based on metadata”
- Data lasts forever, and you might have to someday
  - Data lasts *forever* -- and companies/banks/governments are looking

Alice

½ of Diffie-Hellman → $preA_1$

½ of Diffie-Hellman → $preA_2$

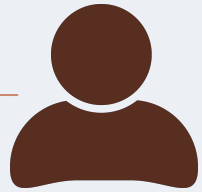
...

Server

½ of Diffie-Hellman → $preB_1$

½ of Diffie-Hellman → $preB_2$

...



Bob

$preB_1$ → Diffie-Hellman → sk_1

Encrypt(m_1, sk_1) → c_1

c_1

Rest of Diffie-Hellman → sk_1

Ratchet forward sk_1 → sk_2

Ratchet forward sk_1 → sk_2

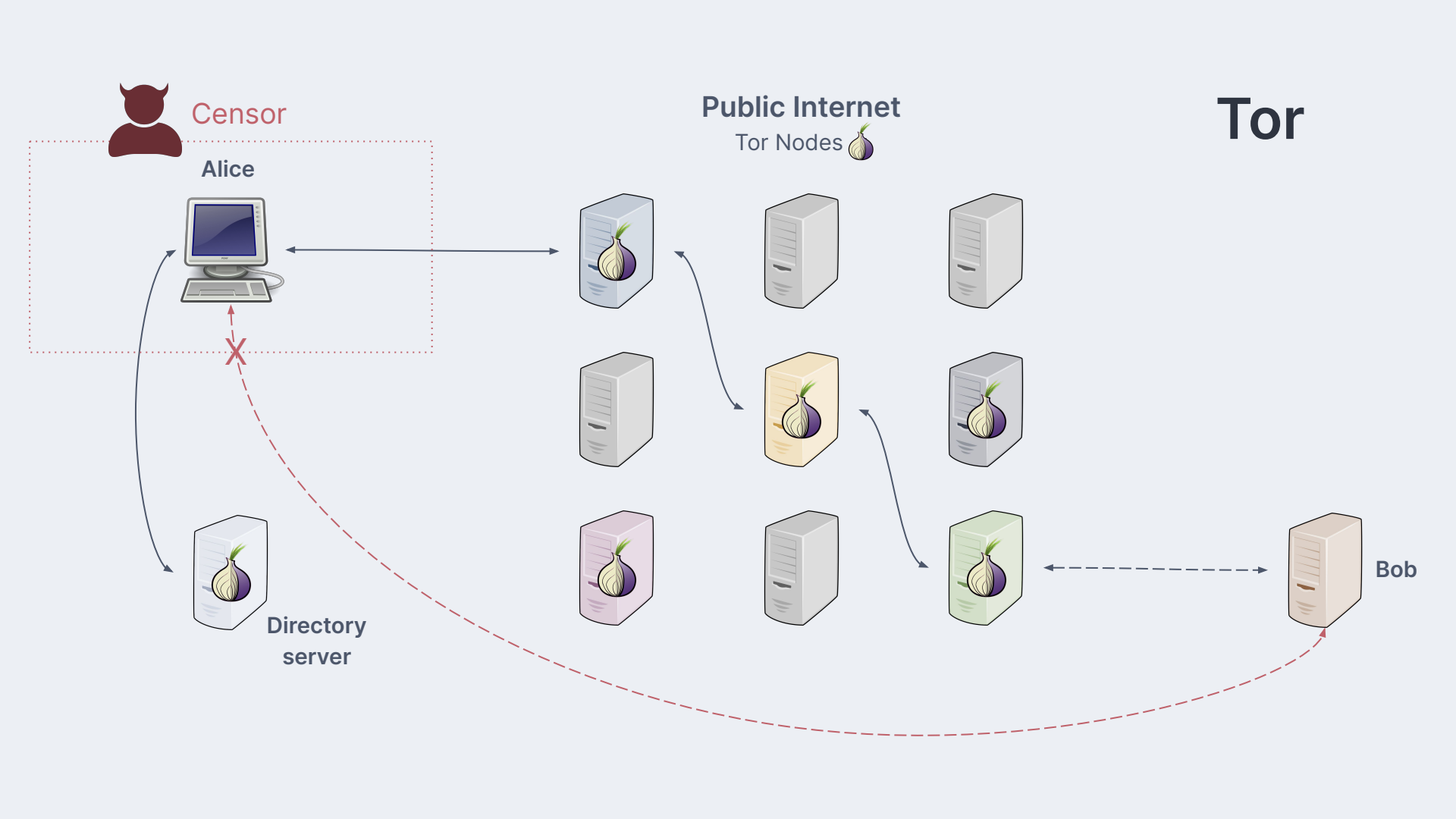
c_2

Encrypt(m_2, sk_2) → c_2

Ratchet forward sk_2 → sk_3

Ratchet forward sk_2 → sk_3

The loss of one key doesn't leak previous ones → forward secrecy


Censor
Alice
Public Internet
Tor Nodes
Tor
Directory server
Bob

da·ta-o·bliv·i·ous com·pu·ta·tion (*n.*)

a program execution with the same observable characteristics regardless of the inputs provided

see also constant-time programming
antonym data-dependent computation

?

Looking ahead

- Key dates for the remainder of the semester
  - Apr 15: Project code due
  - Apr 16: Exam 2
  - May 7: Project demo day
  - May 14: Project presentations
- Do you want your Exam 1 back?
  - Send me an email with subject “EE G7701 Exam 1 [LastName]” from your CCNY address
- **Today’s activity**: Project Check-In 3

Lesson objectives

- Understand the problems of outsourcing scientific computation securely
- Apply data-oblivious computation to address microarchitectural side-channel attacks
- Be prepared for the second midterm exam in the course