EXERCISE #14

REVIEW: INFORMATION FLOW

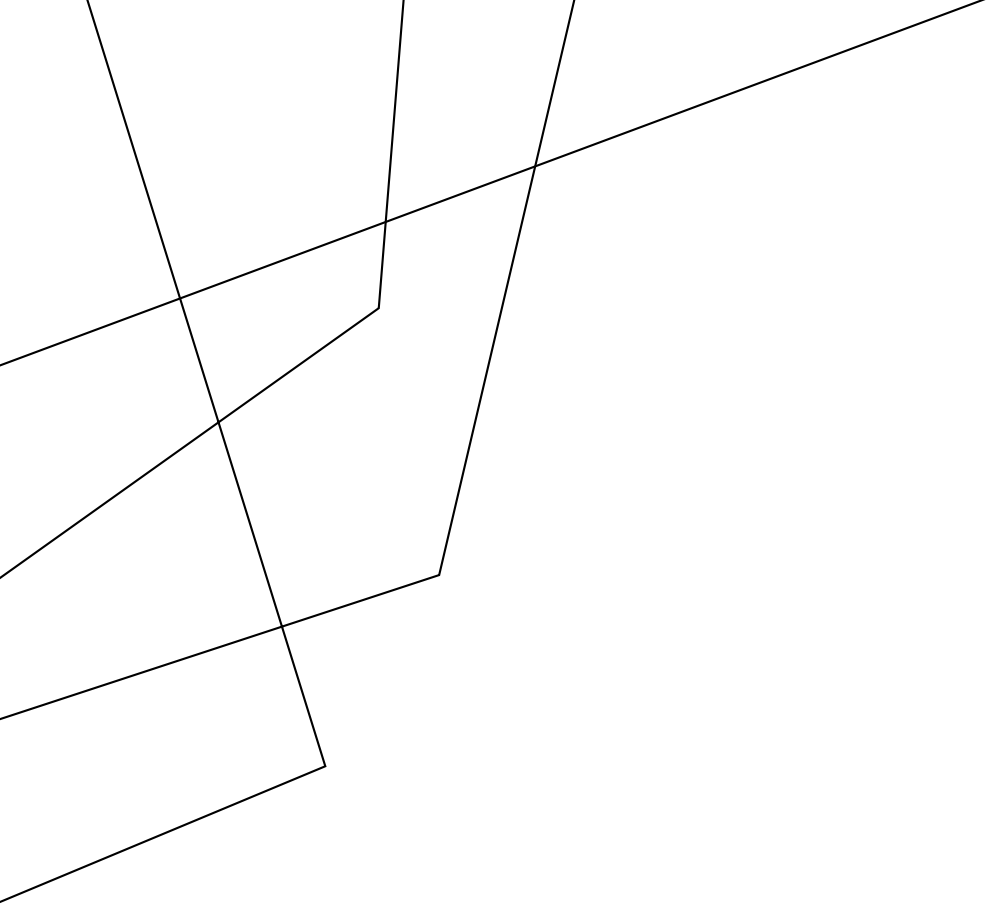
Write your name and answer the following on a piece of paper

Provide an instance of a program with an implicit information flow from a confidential (high-security) source to an untrusted (low-security) sink

EXERCISE #14: SOLUTION

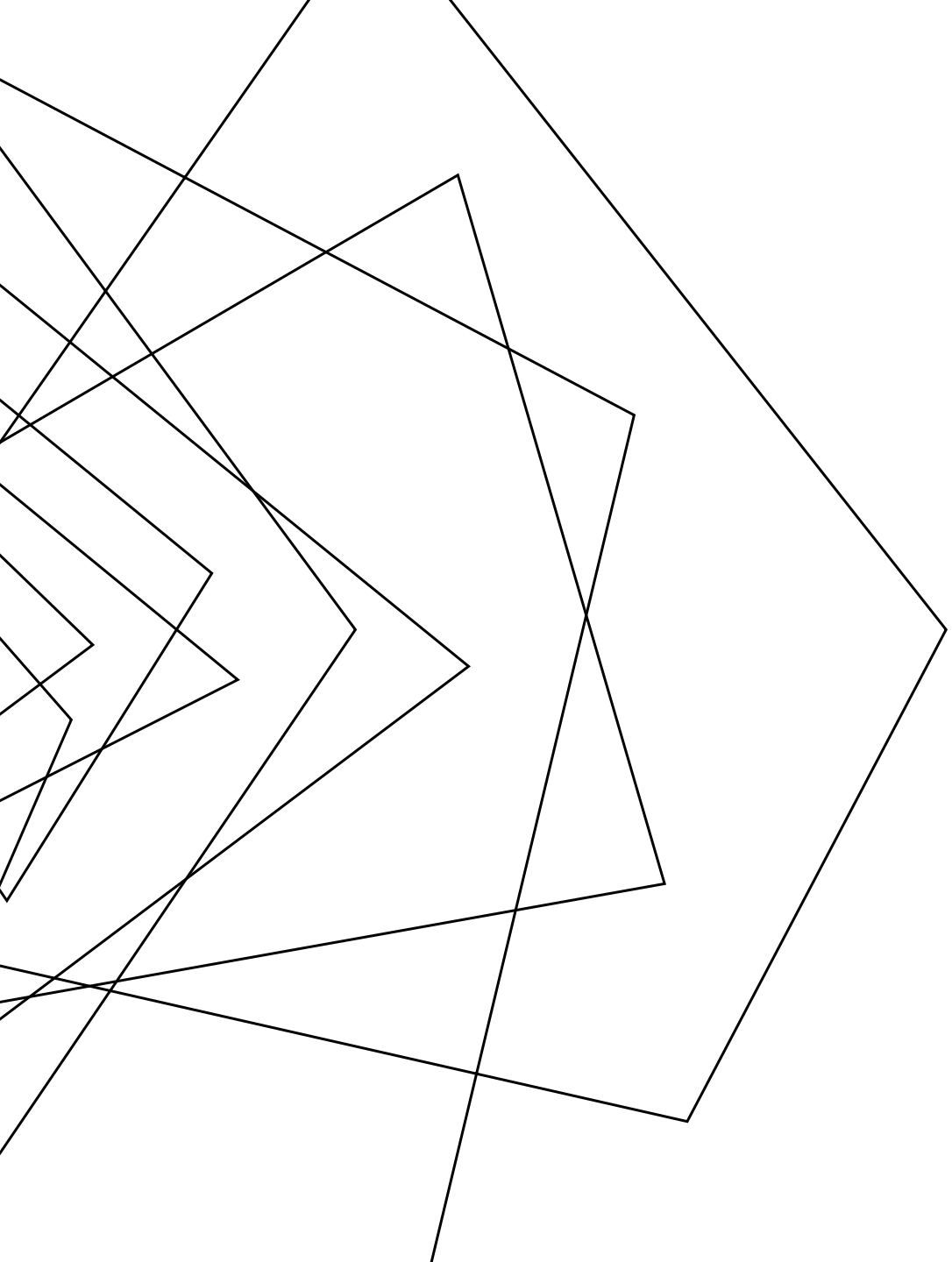
REVIEW: INFORMATION FLOW

Provide an instance of a program with an implicit information flow from a confidential (high-security) source to an untrusted (low-security) sink



ADMINISTRIVIA AND ANNOUNCEMENTS

For EECS777 Students: Paper #1 reading assignment is up!



CLASS PROGRESS

SHOWING SOME APPLICATIONS OF STATIC DATAFLOW

LAST TIME: DATAFLOW DEPLOYMENT

PREVIOUSLY: INFORMATION FLOW

USING DATAFLOW IN PRACTICAL CONTEXTS

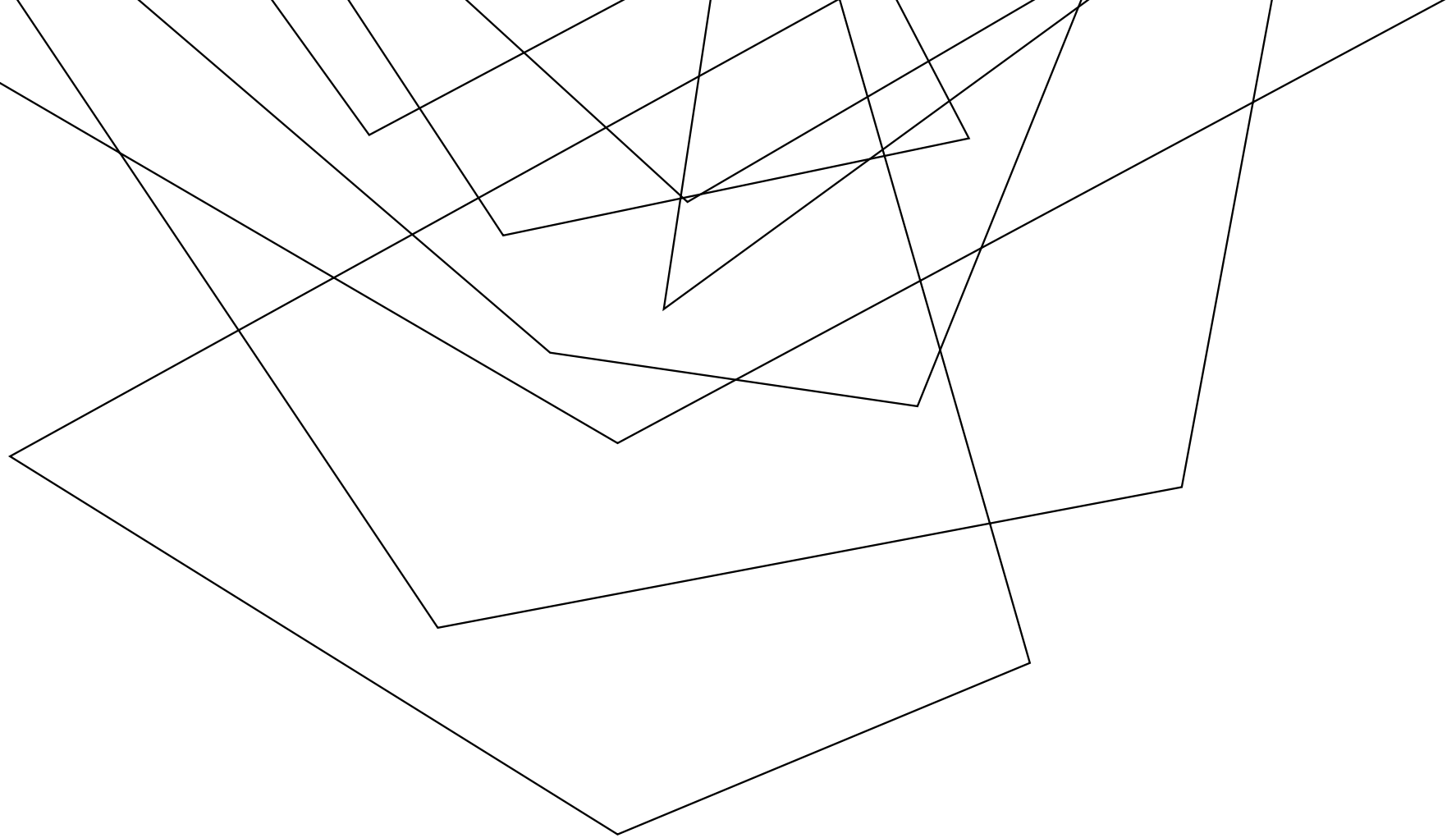
- Ex. - Looking for secret-holding variables

IMPLICIT FLOW

PREVIOUSLY: INFORMATION FLOW

```
X = source();
Y = 0;
if (X == 1){
  Y = 1;
  SINK(1);
}
if (X == 2){
  Y = 2;
}
…
```

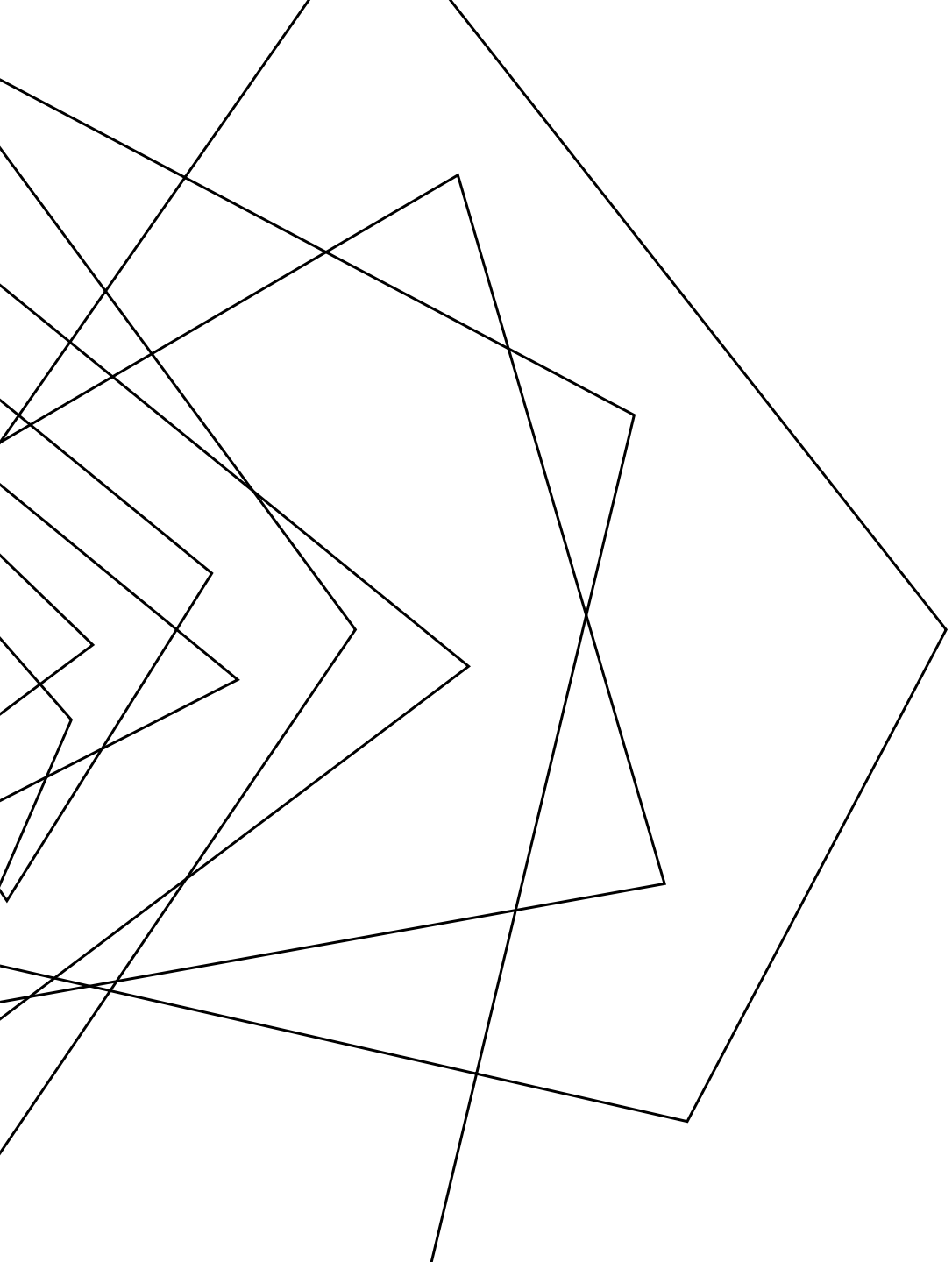
Leak!



SIDE CHANNELS

EECS 677: Software Security Evaluation

Drew Davidson

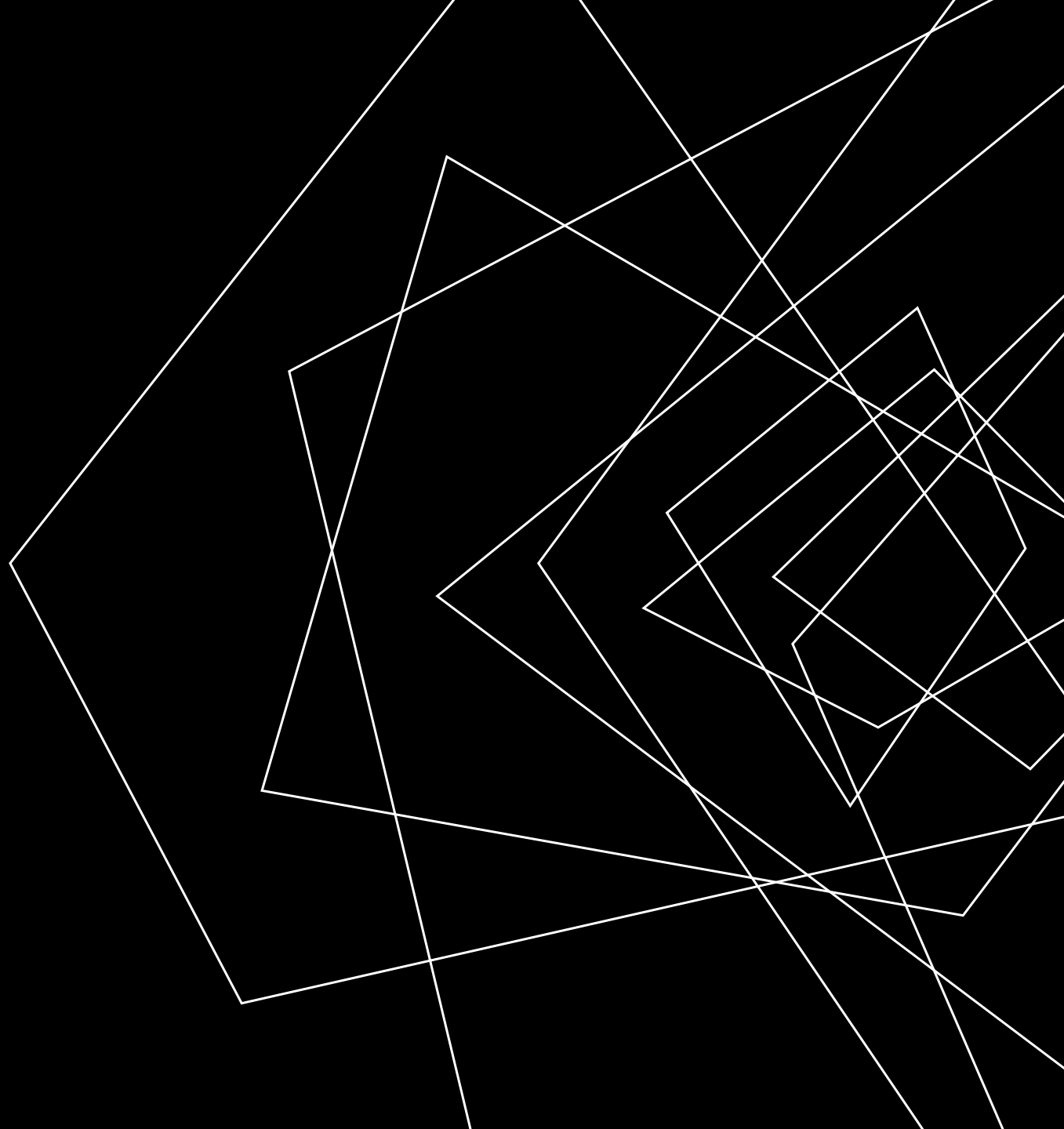


OVERVIEW

CONTEMPLATE OTHER WAYS THAT SNEAKY DATA FLOWS CAN OCCUR

LECTURE OUTLINE

- Threat Models
- Side Channels - Overview
- Timing
- A dataflow approach



THINKING ABOUT ATTACKS

RECALL: THREAT MODELS

THERE'S NO SUCH THING AS "ABSOLUTE SECURITY"

- It's always possible to come up with SOME (potentially wacky) scenario where the adversary can subvert a system

CONSIDER THE VARIOUS ATTACK CLASSES

- **Denial of Service:** Availability is compromised
- **Exfiltration:** Confidentiality policy is compromised
- **Compromise:** Integrity policy is compromised

THINKING ABOUT ATTACKS

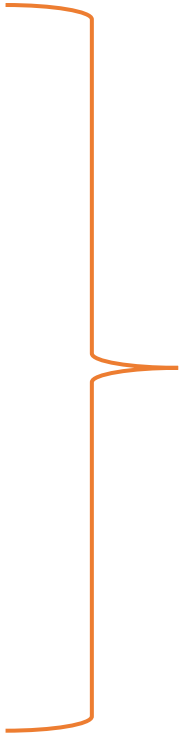
RECALL: THREAT MODELS

THERE'S NO SUCH THING AS "ABSOLUTE SECURITY"

- It's always possible to come up with SOME (potentially wacky) scenario where the adversary can subvert a system

CONSIDER THE VARIOUS ATTACK CLASSES

- **Denial of Service:** Availability is compromised
- **Exfiltration:** Confidentiality policy is compromised
- **Compromise:** Integrity policy is compromised

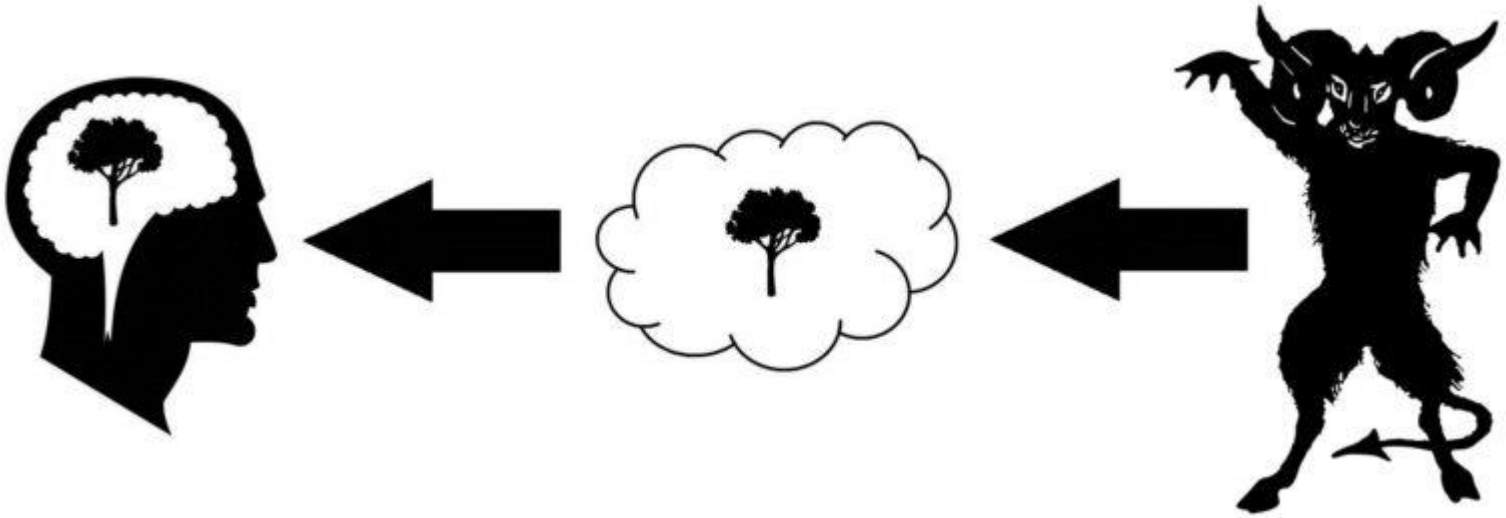


These assumptions are captured by a threat model

UNCONVENTIONAL ADVERSARIES

RECALL: THREAT MODELS

OUR NOTIONS OF COMPLETENESS ARE ULTIMATELY TIED TO OUR ASSUMPTIONS



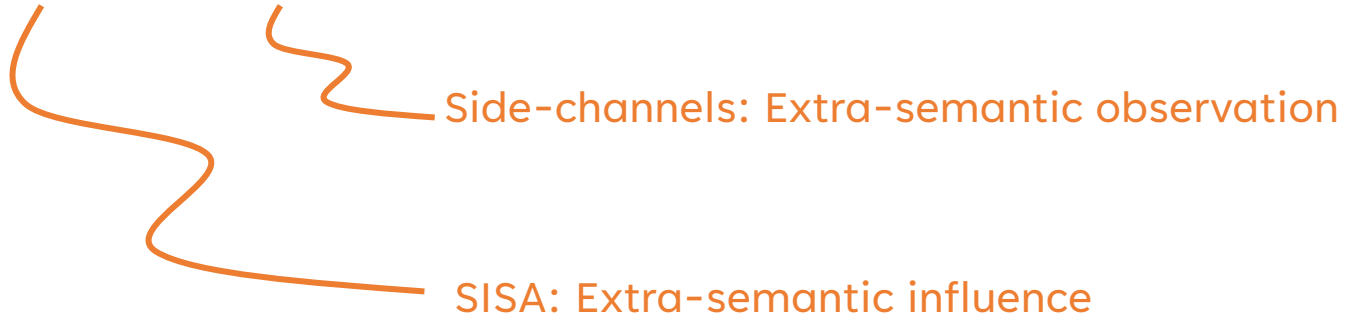
Deus Deceptor (Game over)

UNCONVENTIONAL ADVERSARIES

RECALL: THREAT MODELS

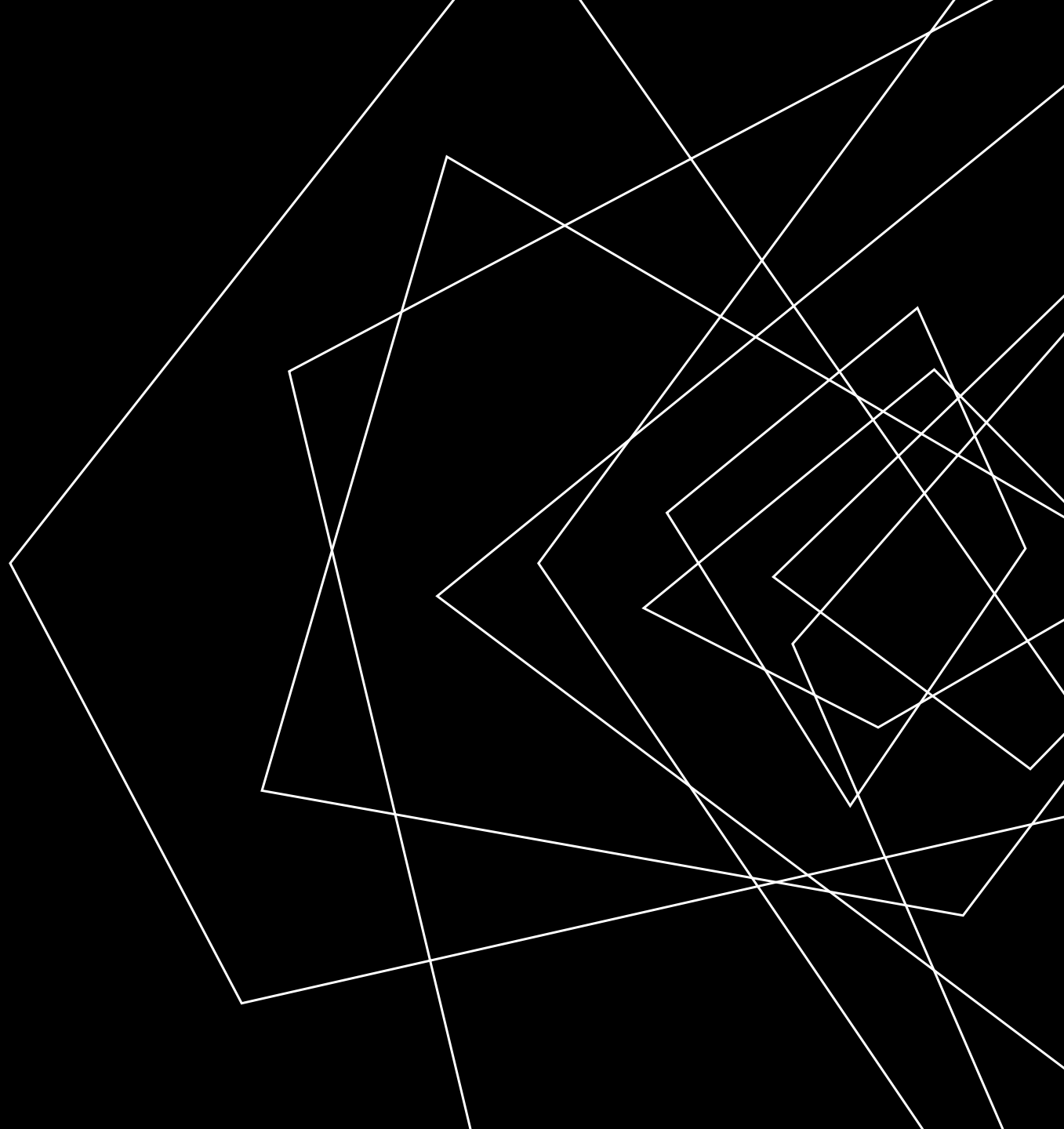
OUR NOTIONS OF COMPLETENESS ARE ULTIMATELY TIED TO OUR ASSUMPTIONS

- An adversary may have the ability to influence (or observe) phenomena that are outside of the threat model
- Anecdote: sensor input spoofing attacks



LECTURE OUTLINE

- Threat Models
- Side Channels - Overview
- Timing
- A dataflow approach



THE BASIC IDEA OF SIDE CHANNELS

Abstraction is a key principle of Computer Science!

As a programmer, you shouldn't need to know underlying details

As a security expert, these details might end up being important!

The way a program accomplishes its tasks are important, especially from a security aspect

- How long does it take for the program to do X ?
- How hot does it make the processor when X happens?
- How much power does it draw when X happens?

UNCONVENTIONAL ADVERSARIES

RECALL: THREAT MODELS

(SADLY) OUR SOFTWARE NEEDS TO BE MANIFESTED IN HARDWARE

SIDE CHANNELS – THE BIG IDEA

SIDE CHANNELS - INSTANCES

COMPUTATION MAY HAVE EFFECTS OUTSIDE OF PROGRAM SEMANTICS

Some operations (internally) take longer based on aspects of the data

TEMPEST

SIDE CHANNELS – HISTORY

Electromagnetic Leakage of keys

- **WWII:** Bell Telephone discovers electromagnetic leakage in one-time pad teleprinters, detectable at 100-ft radius
- **1951:** CIA rediscovers leakage, detectable at 200-ft radius
- **1964:** TEMPEST shielding protocol established

TEMPEST

SIDE CHANNELS – HISTORY

Electromagnetic Leakage of keys

- **WWII:** Bell Telephone discovers electromagnetic leakage in one-time pad teleprinters, detectable at 100-ft radius
- **1951:** CIA rediscovers leakage, detectable at 200-ft radius
- **1964:** TEMPEST shielding protocol established

VAN ECK PHREAKING

SIDE CHANNELS – HISTORY

Electromagnetic Leakage of Monitors

- Pick up the monitor's electromagnetic emanations that differ depending on how the screen lights up
- Originally determined for CRT (1985), also discovered for LCD monitors (2004)

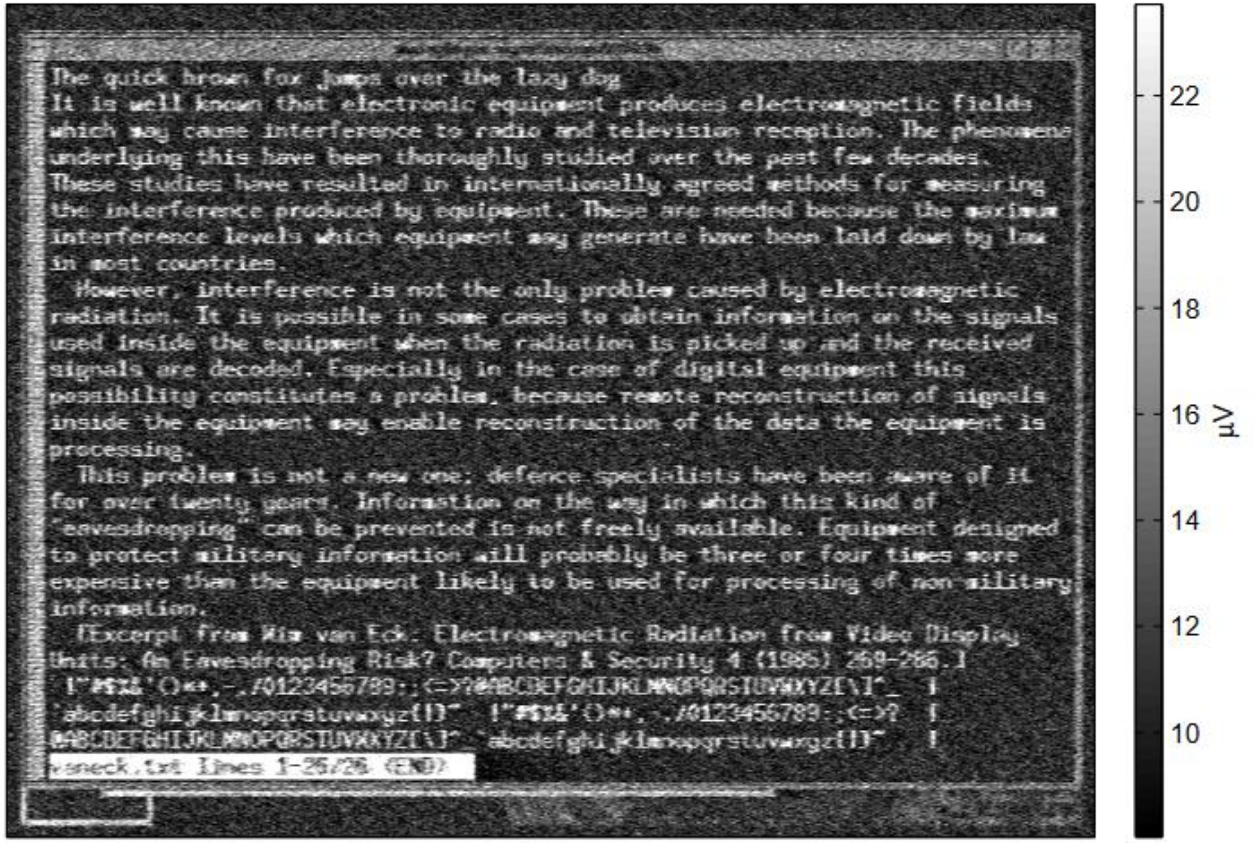


Fig. 3. Text signal received from a 440CDX laptop at 10 m distance through two intermediate offices (3 plasterboard walls).

SIDE CHANNELS – PARTIAL CREDIT

EVEN "HINTS" ABOUT SECRET DATA CAN BE PROBLEMATIC

Assume you're trying to guess a password

– knowing even 1 character massively reduces the search space

– knowing the length of the password reduces the search space

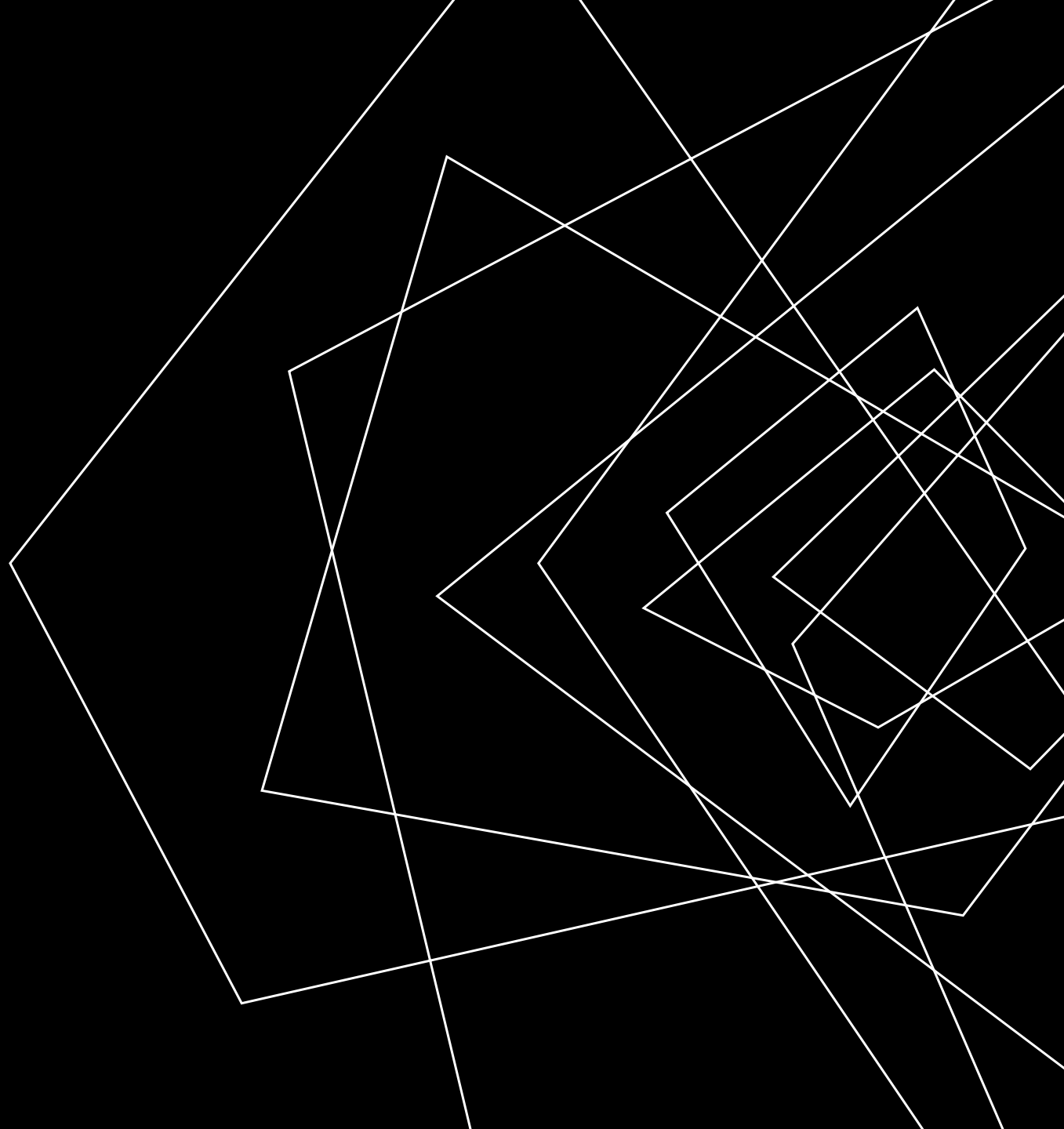
COVERT CHANNELS
SIDE CHANNELS

Sometimes a Program Wants to Leak data

Exfiltration !

LECTURE OUTLINE

- Threat Models
- Side Channels - Overview
- Timing
- A dataflow approach



TIMING SIDE CHANNELS

SIDE CHANNELS - INSTANCES

SOME COMPUTATIONS TAKE LONGER THAN OTHERS

Some operations (internally) take longer based on aspects of the data

```
bool checkPW(const char * given){
  const char * expected = “12345”;
  int gLen = strlen(given);
  int eLen = strlen(expected);
  if (gLen != eLen){ return false; }
  for (int i = 0; i < eLen; i++){
    if (given[i] != expected[i]){
      return false;
    }
  }
  return true;
}
```

TIMING SIDE CHANNELS

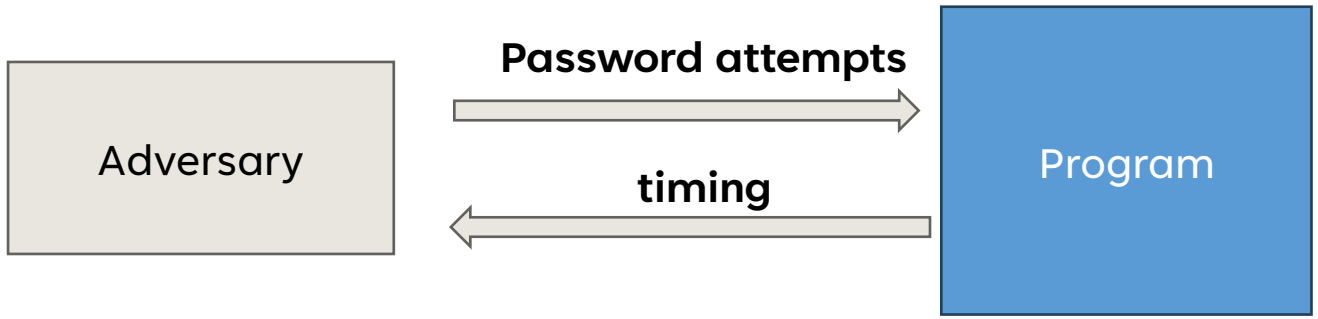
SIDE CHANNELS - INSTANCES

SOME COMPUTATIONS TAKE LONGER THAN OTHERS

Some operations (internally) take longer based on aspects of the data

THREAT MODEL

Interactive, low-latency*, black-box access to the program, precise timer



*: May be overcome with more samples

TIMING SIDE CHANNELS - FIX

SIDE CHANNELS - INSTANCES

```
bool checkPW(const char * given){
  const char * expected = "12345";
  int gLen = strlen(given);
  int eLen = strlen(expected);
  if (gLen != eLen){ return false; }
  for (int i = 0; i < eLen; i++){
    if (given[i] != expected[i]){
      return false;
    }
  }
  return true;
}
```

```
bool checkPW(const char * given){
  const char * expected = "12345";
  int gLen = strlen(given);
  int eLen = strlen(expected);
  bool ok = true;
  if (gLen != eLen){ ok = false; }
  for (int i = 0; i < eLen; i++){
    int gIdx = math.min(gLen - 1, i);
    if (given[gIdx] != expected[i]){
      ok = false;
    }
  }
  return ok;
}
```

TIMING SIDE CHANNELS - FIX

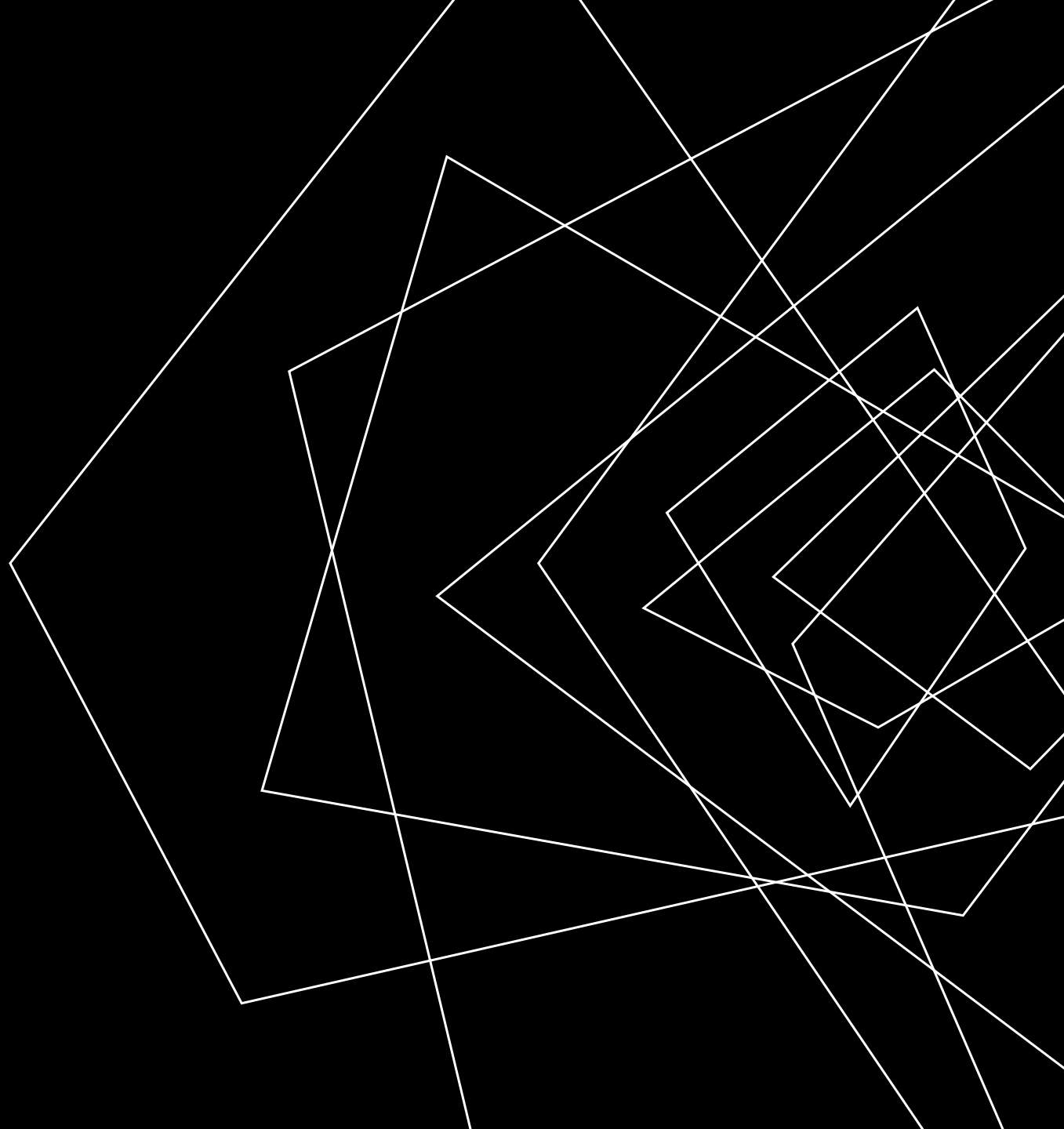
SIDE CHANNELS - INSTANCES

Limitations of Uniform Execution

- Necessarily slow down your computation to the worst case
- May require some pretty precise understanding of timing
- May not always be obvious what the worst-case even is

LECTURE OUTLINE

- Threat Models
- Side Channels - Overview
- Instances
- A dataflow approach



TIMING SIDE CHANNELS - FIX

SIDE CHANNELS - INSTANCES

CAN WE FIX THIS ISSUE WITH OUR DATAFLOW APPROACH?

- Instruction transformers: how much time that instruction takes
- Block composition: the sum total of instruction times
- Merge operation: some sort of check that all paths are of comparable time?

WRAP-UP