

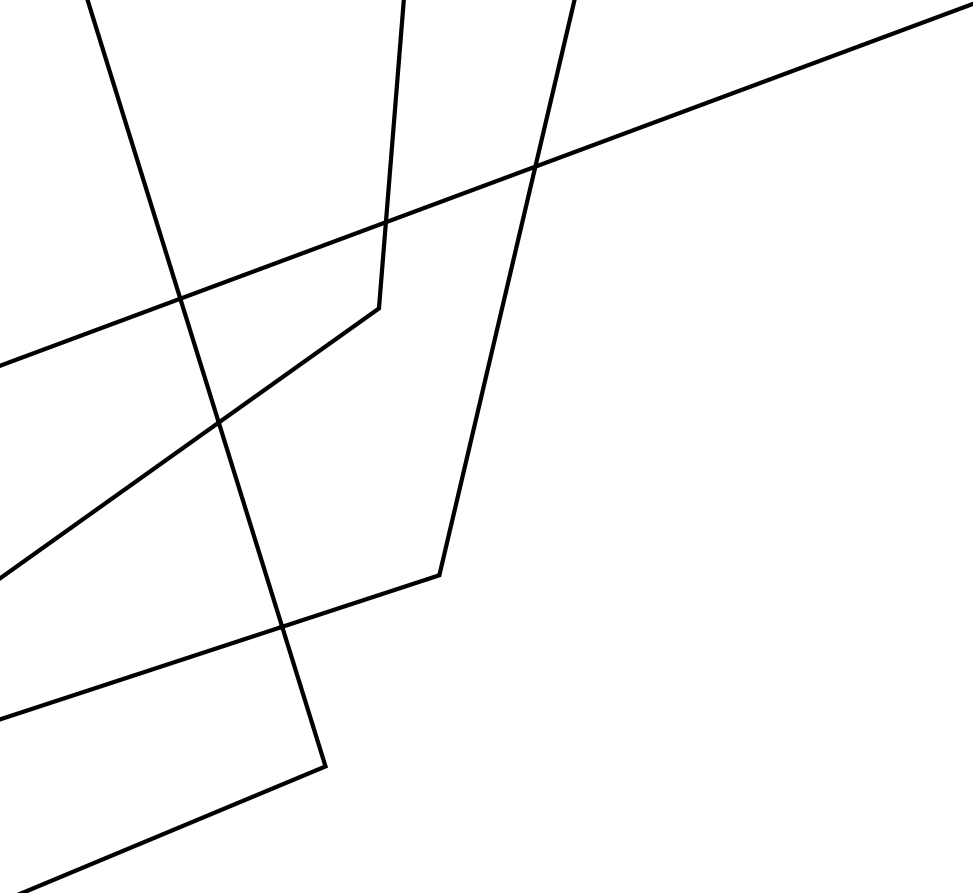
EXERCISE #12

LLVM CALL REVIEW

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

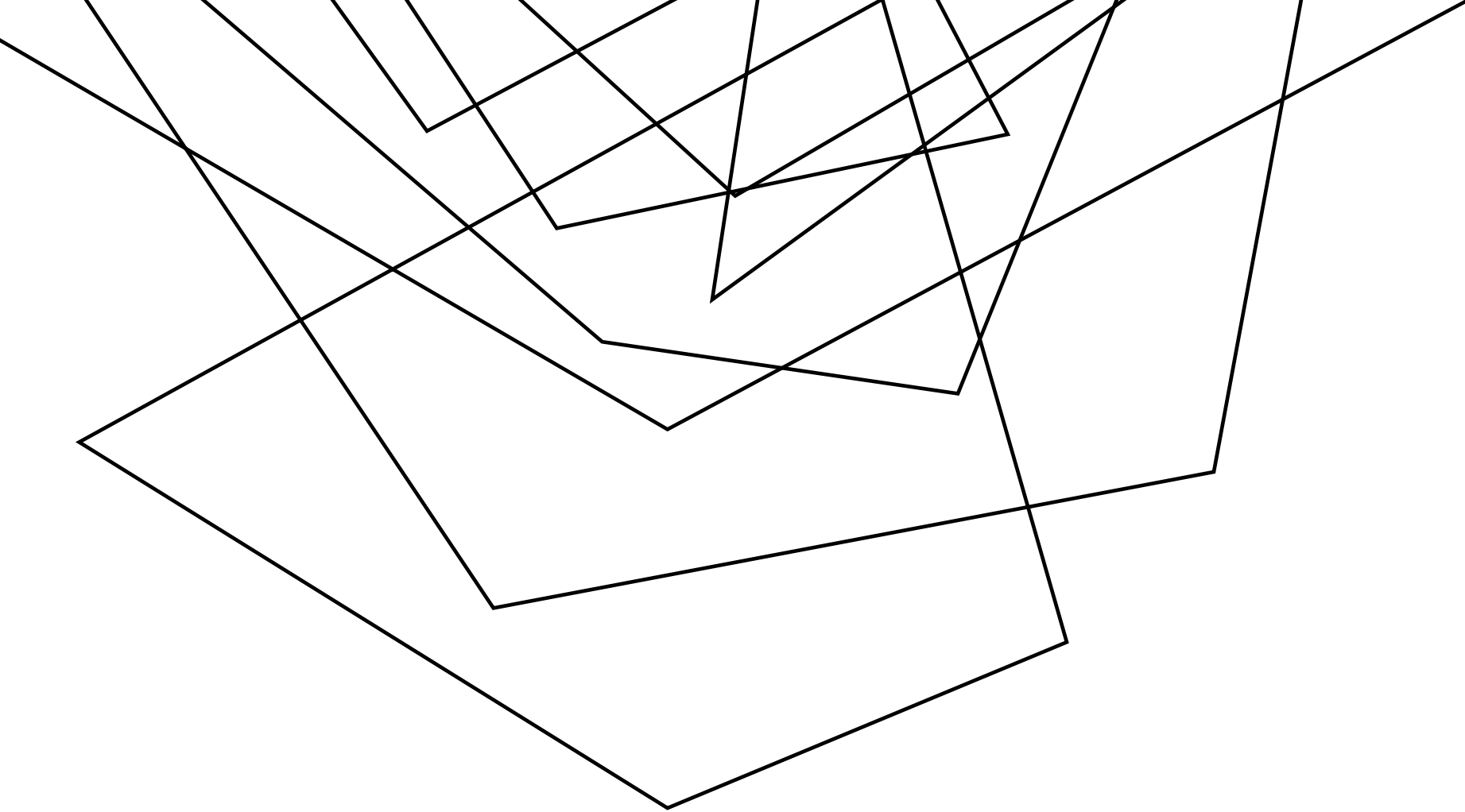
- *What does this program do?*

```
@.str = constant [11 x i8] c"%d skidoo\0A\00", align 1
define i32 @main() #0 {
    %strPtr = getelementptr [11 x i8], [11 x i8]* @.str, i64 0, i64 0
    %reg = call i32 (i8*, ...) @printf(i8* %strPtr, i32 23)
    ret i32 %reg
}
```



Hope you enjoyed the career fair!

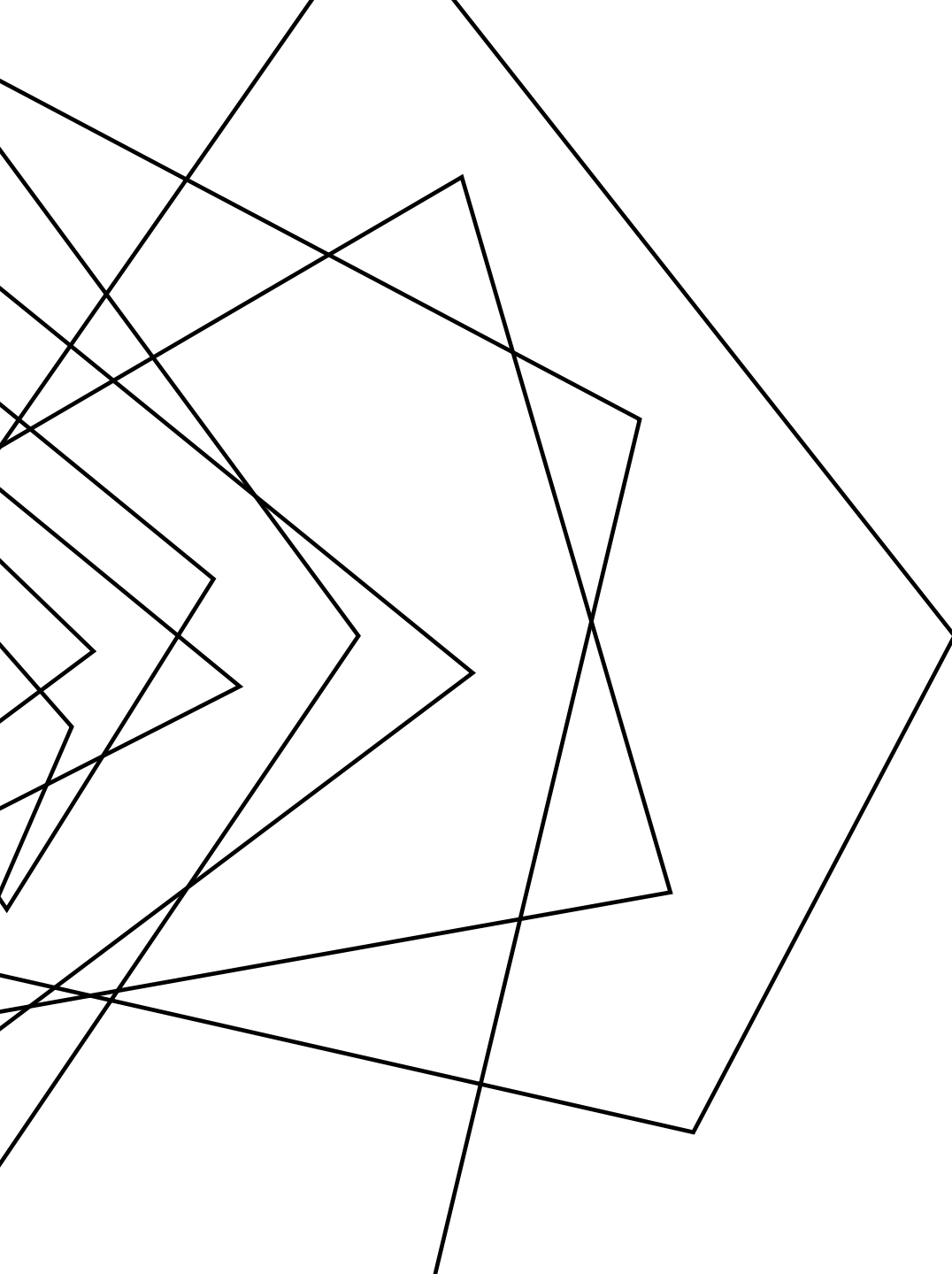
**ADMINISTRIVIA
AND
ANNOUNCEMENTS**



INFORMATION FLOW

EECS 677: Software Security Evaluation

Drew Davidson



CLASS PROGRESS

BUILT UP THE THEORY AND PRACTICE OF
STATIC DATAFLOW

- TIME TO PUT THEM TO USE!

LAST TIME: LLVM CALLS

REVIEW: LAST LECTURE

DESCRIBED THE SYNTAX OF FUNCTION CALLS

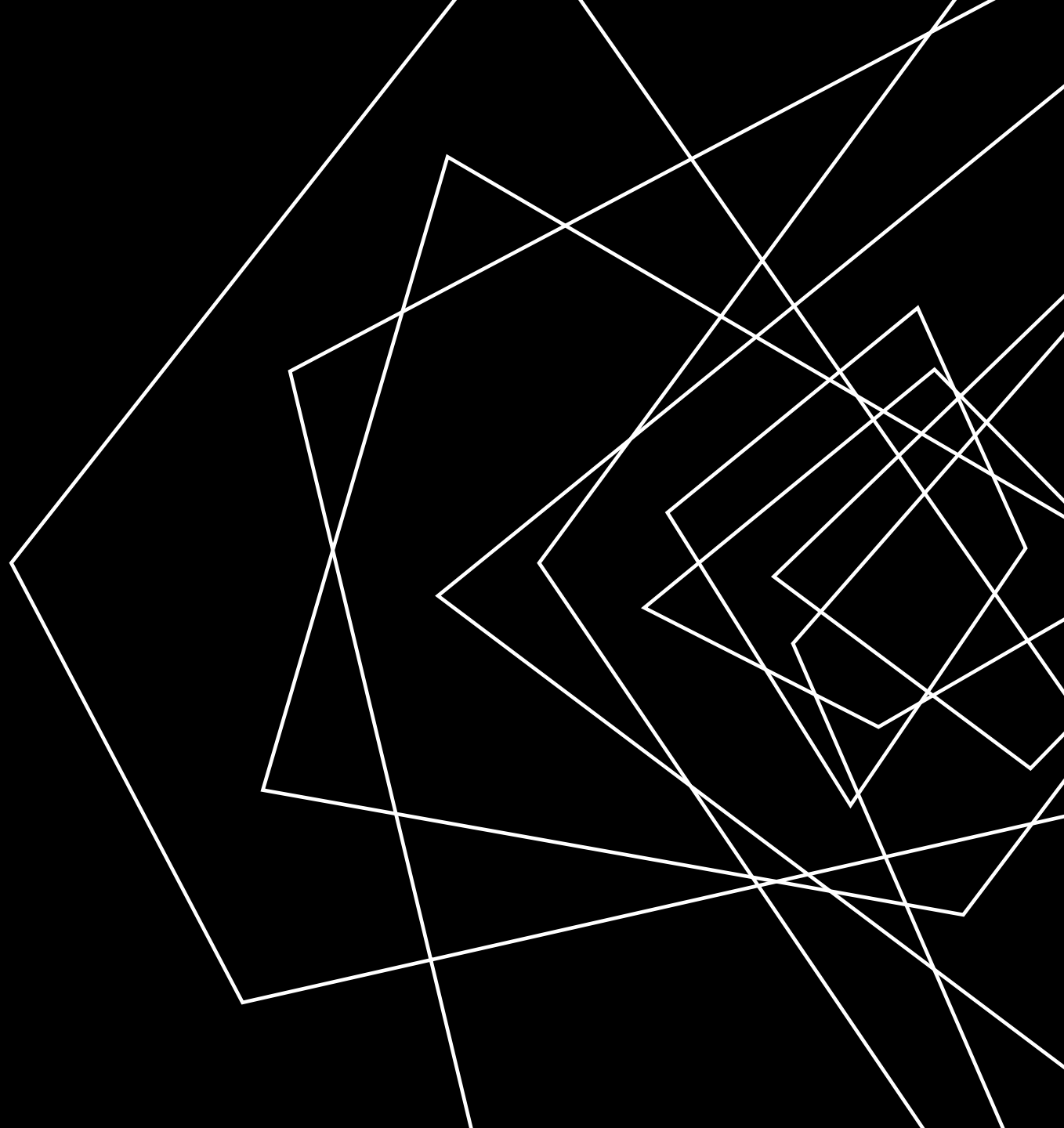
```
call    <callee sig> <function name>    ( <argList> )
%len = call i32 (i8*, ...) @printf(i8* %strPtr, i32 123)
```

```
call    <ret type> <function name>      ( <argList> )
%res = call    i32          @atoi          (i8* %elt1)
```

SHOWED THAT LLVM WILL IMPLICITLY LINK TO C STANDARD LIBRARY FUNCTIONS

LECTURE OUTLINE

- The CIA Triad
- Information flow control
- Information leakage



THE CIA TRIAD

LLVM BITCODE



“THE PRIMARY FOCUS OF INFORMATION SECURITY” - WIKIPEDIA

Confidentiality – the control of access to data

Integrity – the consistency, accuracy and trustworthiness of data over its entire lifecycle

Availability – The degree of consistent accessibility of data

THE CIA TRIAD

LLVM BITCODE

“THE PRIMARY FOCUS OF INFORMATION SECURITY” - WIKIPEDIA

*(imperfect) formulations
as dataflow properties*

Confidentiality – the control of access to data

Integrity – the consistency, accuracy and trustworthiness of data over its entire lifecycle

Availability – The degree of consistent accessibility of data

THE CIA TRIAD

LLVM BITCODE

“THE PRIMARY FOCUS OF INFORMATION SECURITY” - WIKIPEDIA

Confidentiality – the control of access to data

Integrity – the consistency, accuracy and trustworthiness of data over its entire lifecycle

Availability – The degree of consistent accessibility of data

*(imperfect) formulations
as dataflow properties*

Sensitive data from within the program touching an untrusted destination

An untrusted source touching a sensitive destination

INFORMATION CONTROL

INFORMATION FLOW

WHAT ABOUT THE HEAP OF EXISTING SECURITY MECHANISMS?

Firewalls – Block certain types of inbound data

Access control lists / Access control enforcement – bidirectional blocking of certain classes of data

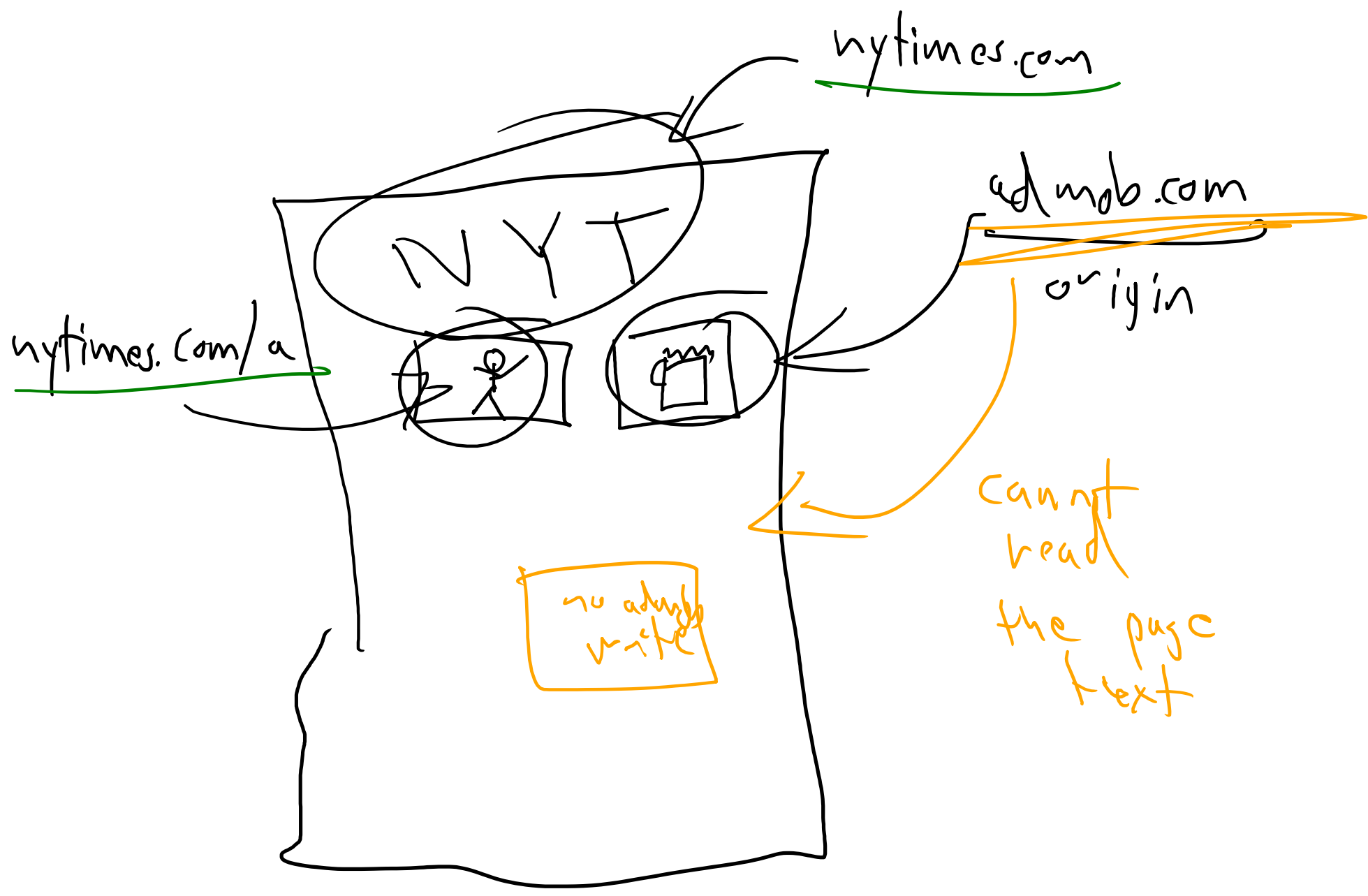
Process isolation – Prohibits data integrity interference between programs

The same origin policy – Prevents communication / influence between entities within a single web app

... many others

THESE MECHANISMS ARE FAIRLY COARSE-GRAINED

Little understanding of the program's behavior



THE SEMANTIC GAP

INFORMATION FLOW

GENERIC DEFINITION

The semantic gap - The difference between descriptions of an object by different linguistic representations

*For CS, focus
on symbolic /
algorithmic
representations
(often of programs)*



THE SEMANTIC GAP

INFORMATION FLOW

The semantic gap - The difference between two descriptions of an object by different linguistic representations

WHAT IS A PROGRAM?



THE SEMANTIC GAP

INFORMATION FLOW

The semantic gap - The difference between descriptions of an object by different linguistic representations


WHAT IS A PROGRAM?

A miserable little pile of
secrets
- Dracula

A sequence of transformations
over memory configurations
- Hardware view

A memory region and a set of
privileges
- OS view

← True but not very helpful



APPLICATION-LEVEL ANALYSIS

INFORMATION FLOW

FOCUS ON THE BEHAVIOR / SEMANTICS OF THE PROGRAM

(Hopefully) the “right” level of granularity for understanding a program’s security

Not the only
option

Definite
limitations

LANGUAGE-BASED SECURITY

INFORMATION FLOW

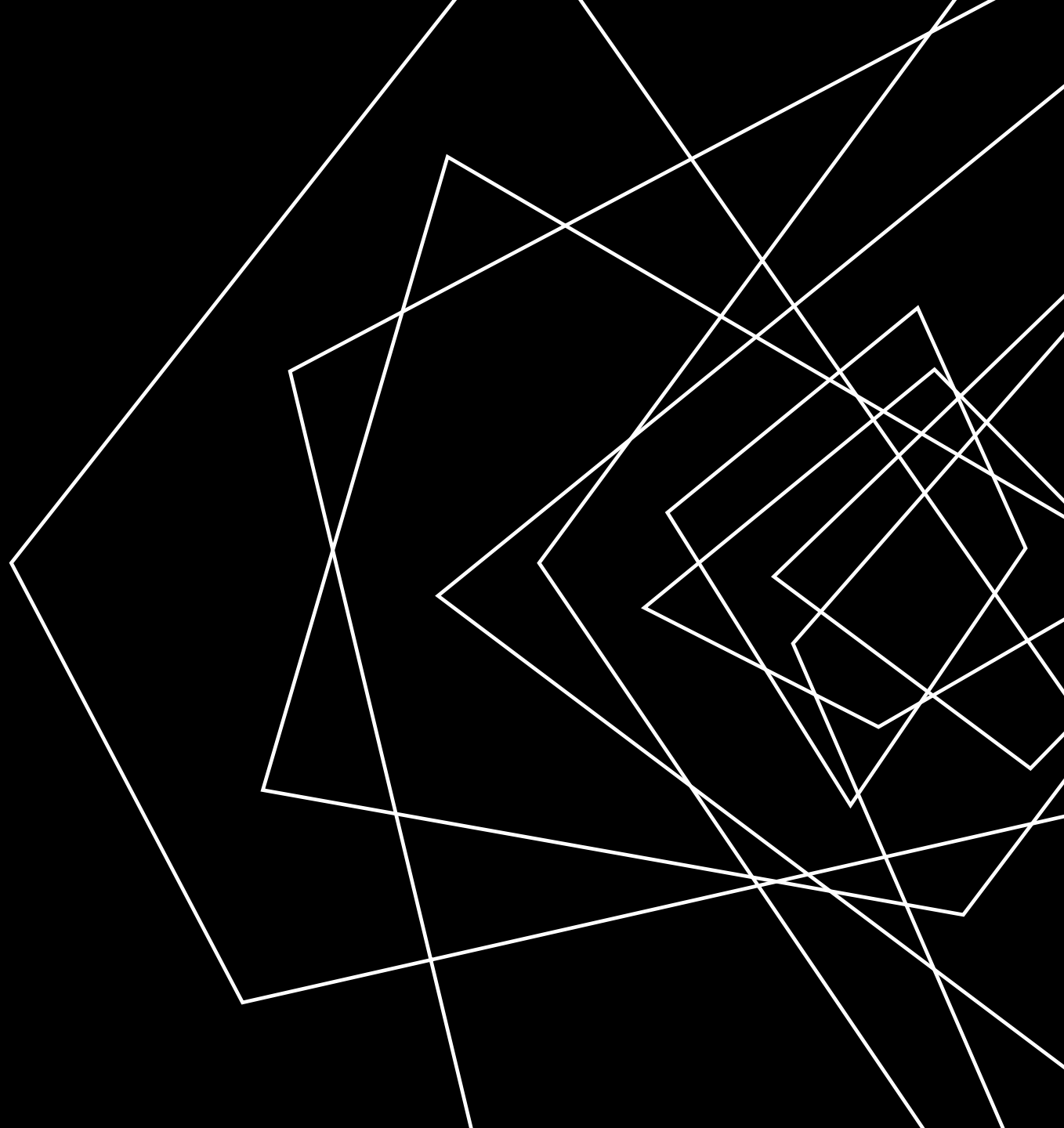
DEFINITION

Language-based security - a set of techniques to strengthen the security of applications by using the properties of programming languages

“Hey, we’ve got all of these great tools to understand programs for the sake of correctness / optimization, they’d work for security too!”

LECTURE OUTLINE

- The CIA Triad
- Information flow control
- Information leakage



INFORMATION FLOW CONTROL

INFORMATION FLOW

SIMPLEST FORMULATION

Divide program data and functionality into “high security” and “low security”

Integrity: low security data should not influence “high security” functionality

Confidentiality: “high security” data should not influence “low security” functionality

INFORMATION FLOW CONTROL

INFORMATION FLOW

LEVERAGING DATAFLOW

Consider a **flow** to be a *program path* segment that begins at a **source node** and ends at a **sink node**

A source node – generates data

A sink node – consumes data

$p: 1, [2, [3, 4]^*], 5$
 program path

$t: 1, 2, 3, 4, 3, 4, 5$

How to mark these nodes depends on what flow we're interested in detecting

EXAMPLE TIME!

INFORMATION FLOW

Detector for location leaks to the network

- source: point/functionality that collects location
- sink: point/functionality that writes to the network

```

void * l = getLatLon();
void * k = l;
networkWrite(k);
  
```

sources: getLatLon

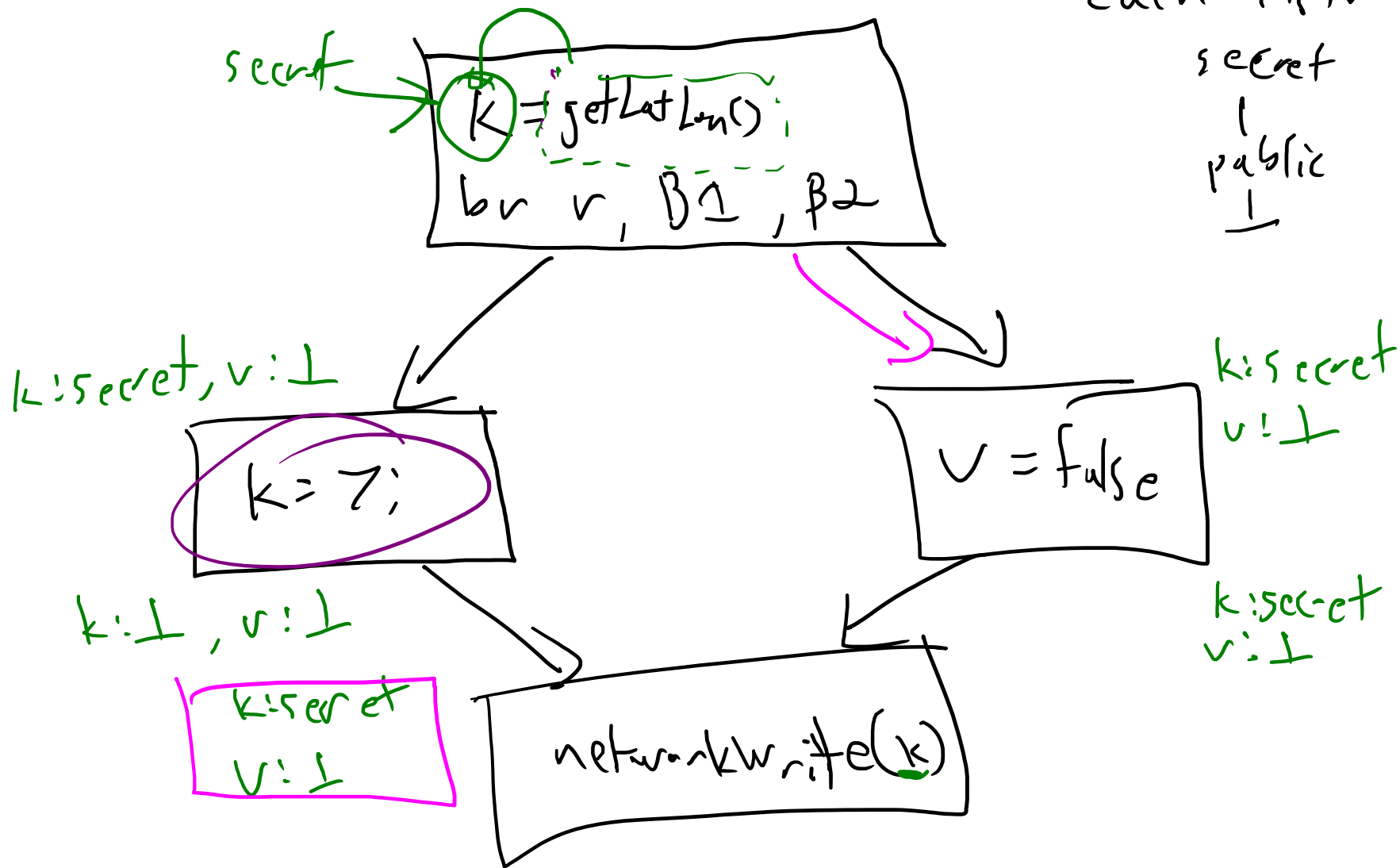
sinks: networkWrite

EXAMPLE TIME!

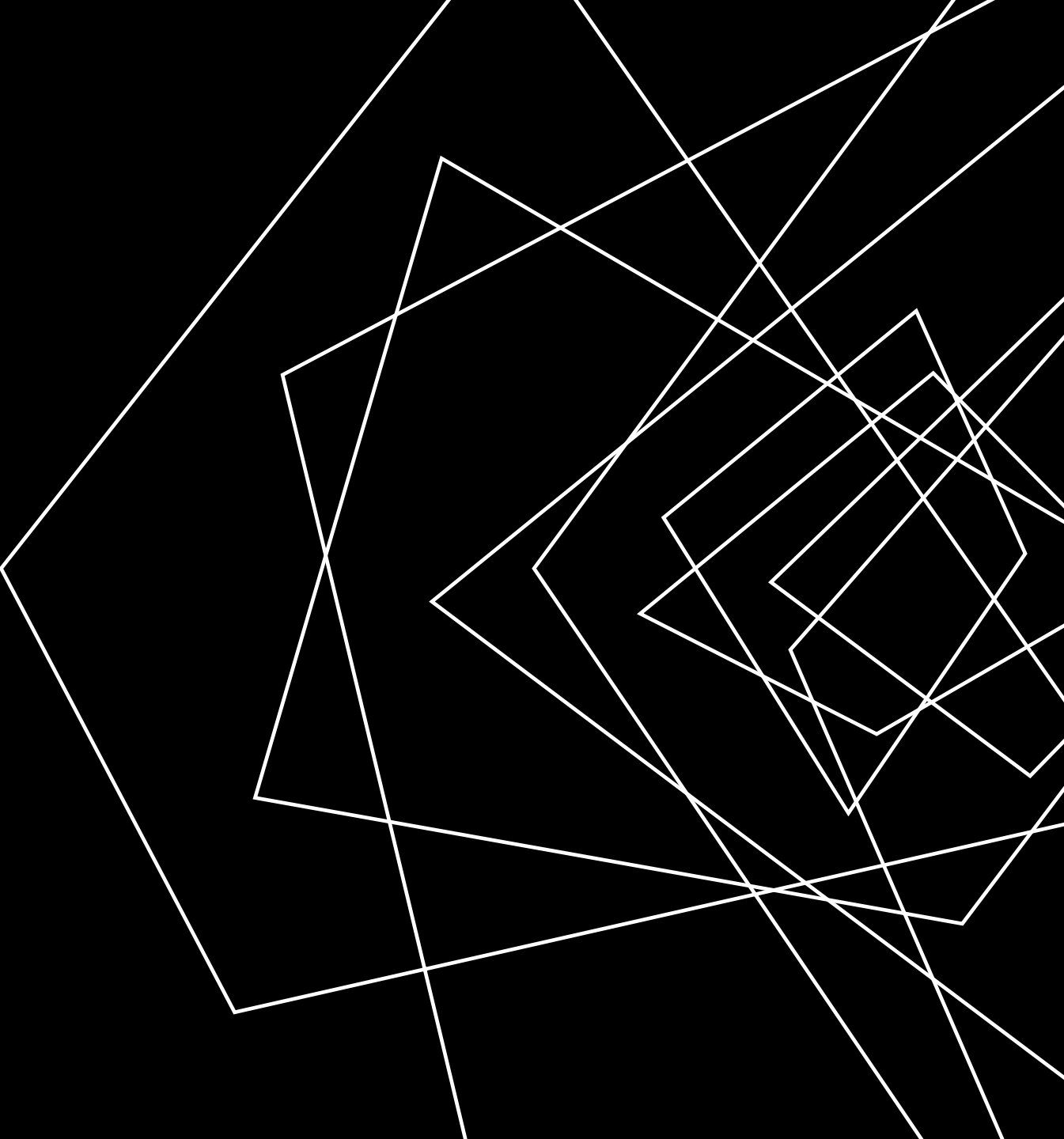
INFORMATION FLOW

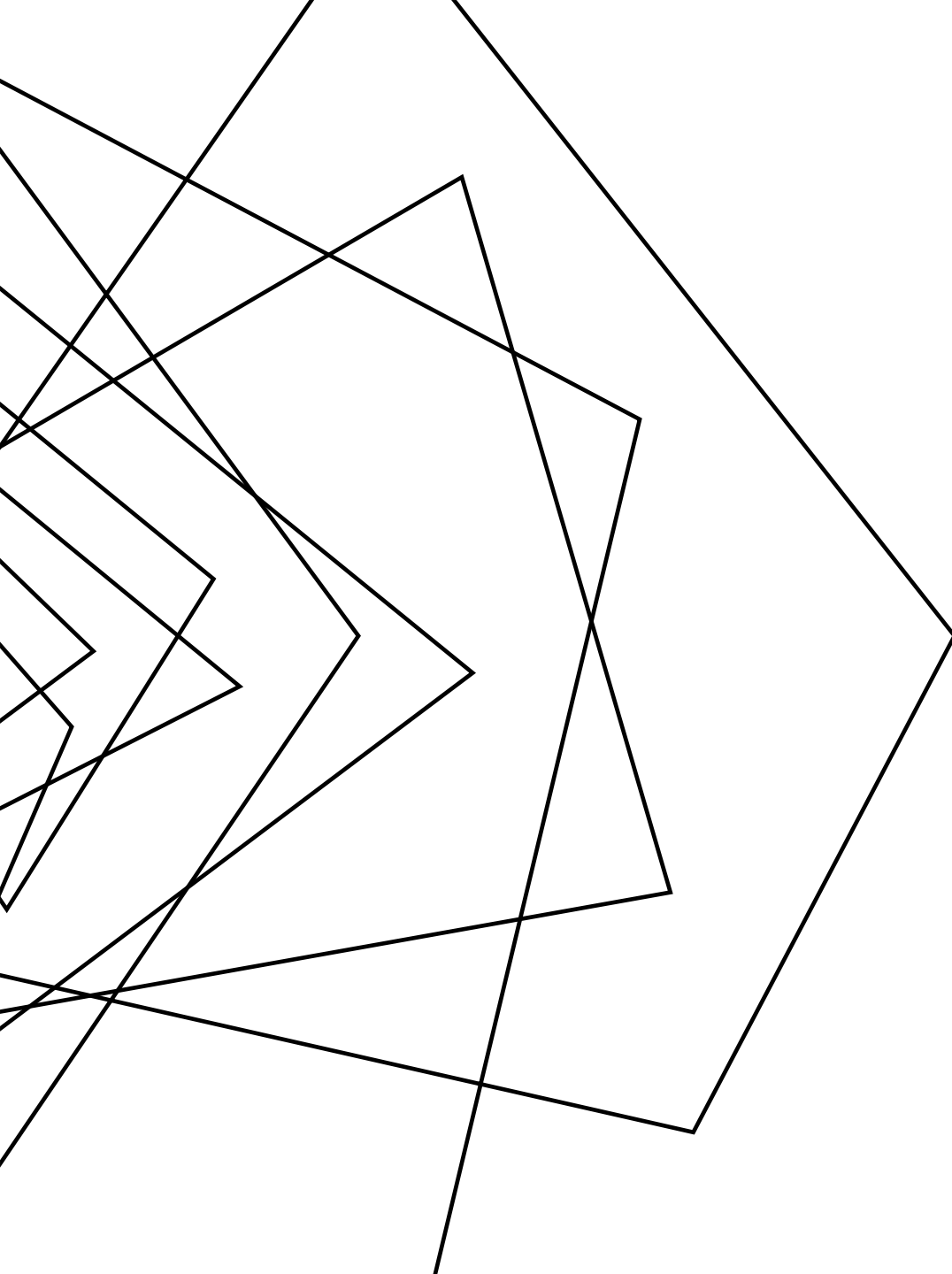
transforms for each instr type!

secret
|
public
|



WRAP-UP





NEXT TIME

WRITING AN ANALYSIS