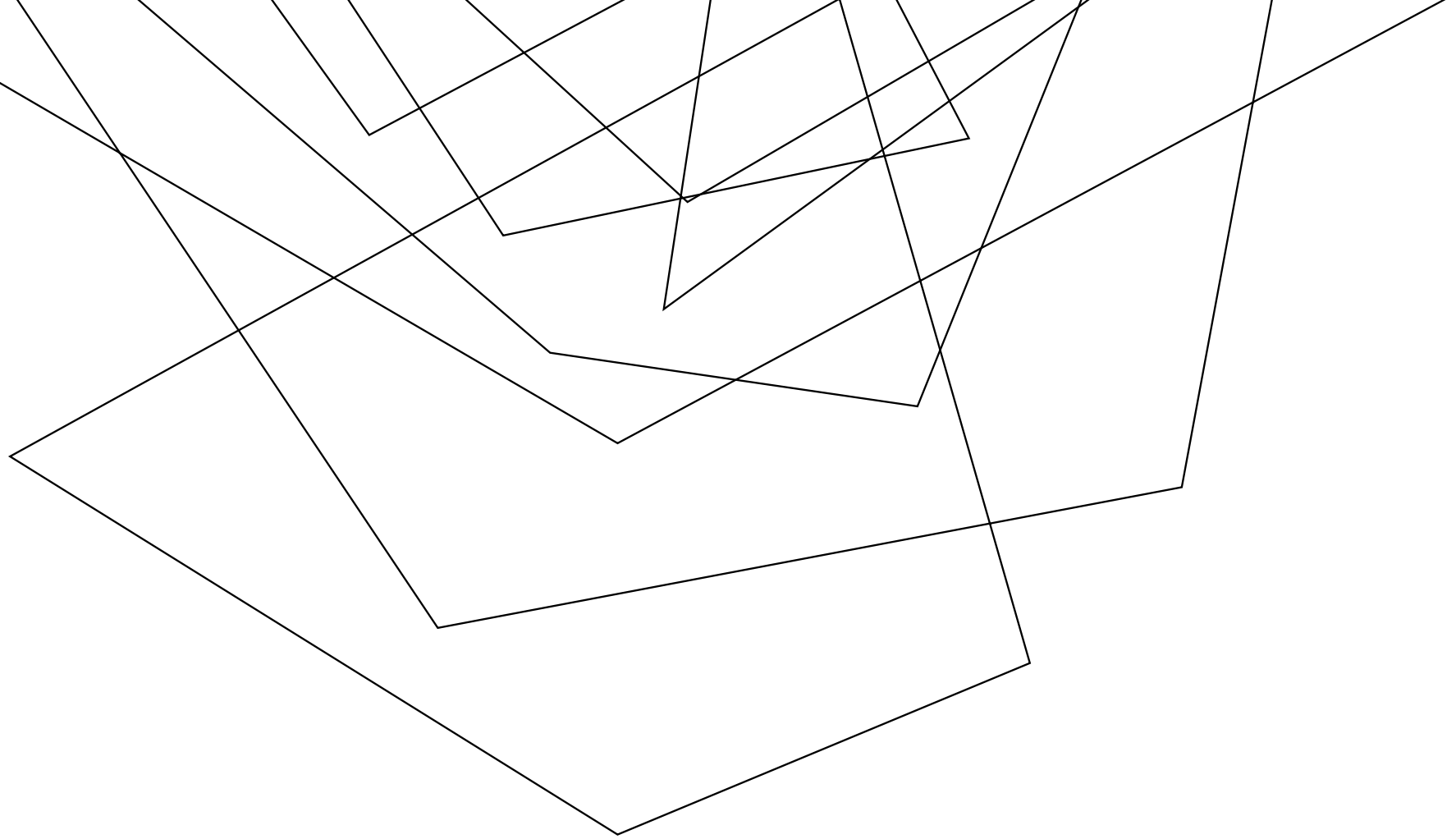


SSDLC

EECS 677: Software Security Evaluation

Drew Davidson

LAST TIME: SMT SOLVING

REVIEW: LAST LECTURE

Beyond pure Boolean reasoning

Theories allow reasoning about a particular set of constants, variables, and operations

Sufficient to capture and reason about many of the constraints generated by the symbolic execution engine

SMT SOLVING – DPLL(T)

REVIEW: LAST LECTURE

CREATE A SATISFYING ASSIGNMENT OF VALUES TO VARIABLES IN CNF FORMULA CLAUSES

Fit clauses into the signature of various theories

May be necessary to re-arranging the formula to separate theories
i.e. replace subformulae with new propositional variables and constraints to separate theories (Nelson-Oppen)

DPLL to determine which clauses to investigate
Create a solution for the high-level clauses
Double check that solution is consistent with theory solvers
If not, block that solution and try again

$$A \wedge (B \vee C) \wedge (\neg D)$$

Where A is $x > 7$

B is $f(x) = 9$

C is $y = 9$

D is $z = 4$

SAYING GOODBYE TO SMT

REVIEW: LAST LECTURE

WE ONLY SCRATCHED THE SURFACE OF SMT SOLVING

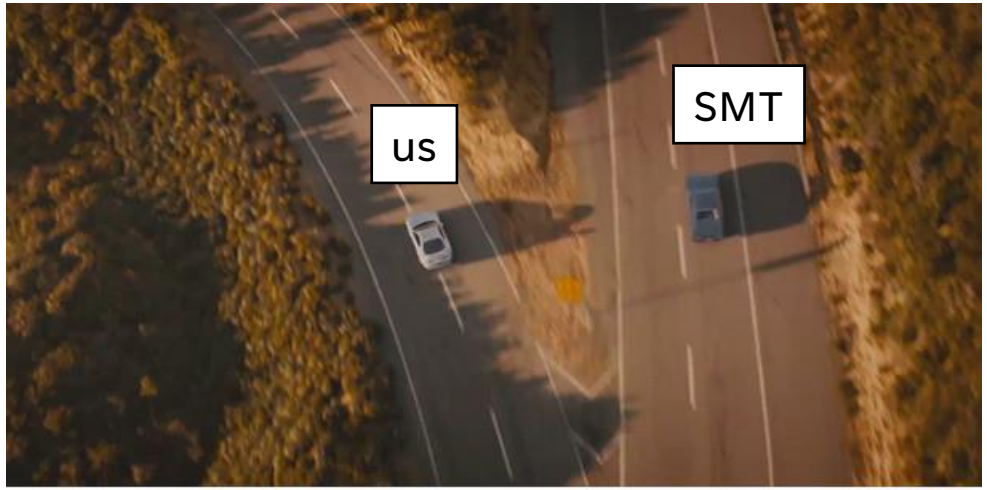
Linear integer arithmetic – use the first phase of the two-phase simplex algorithm

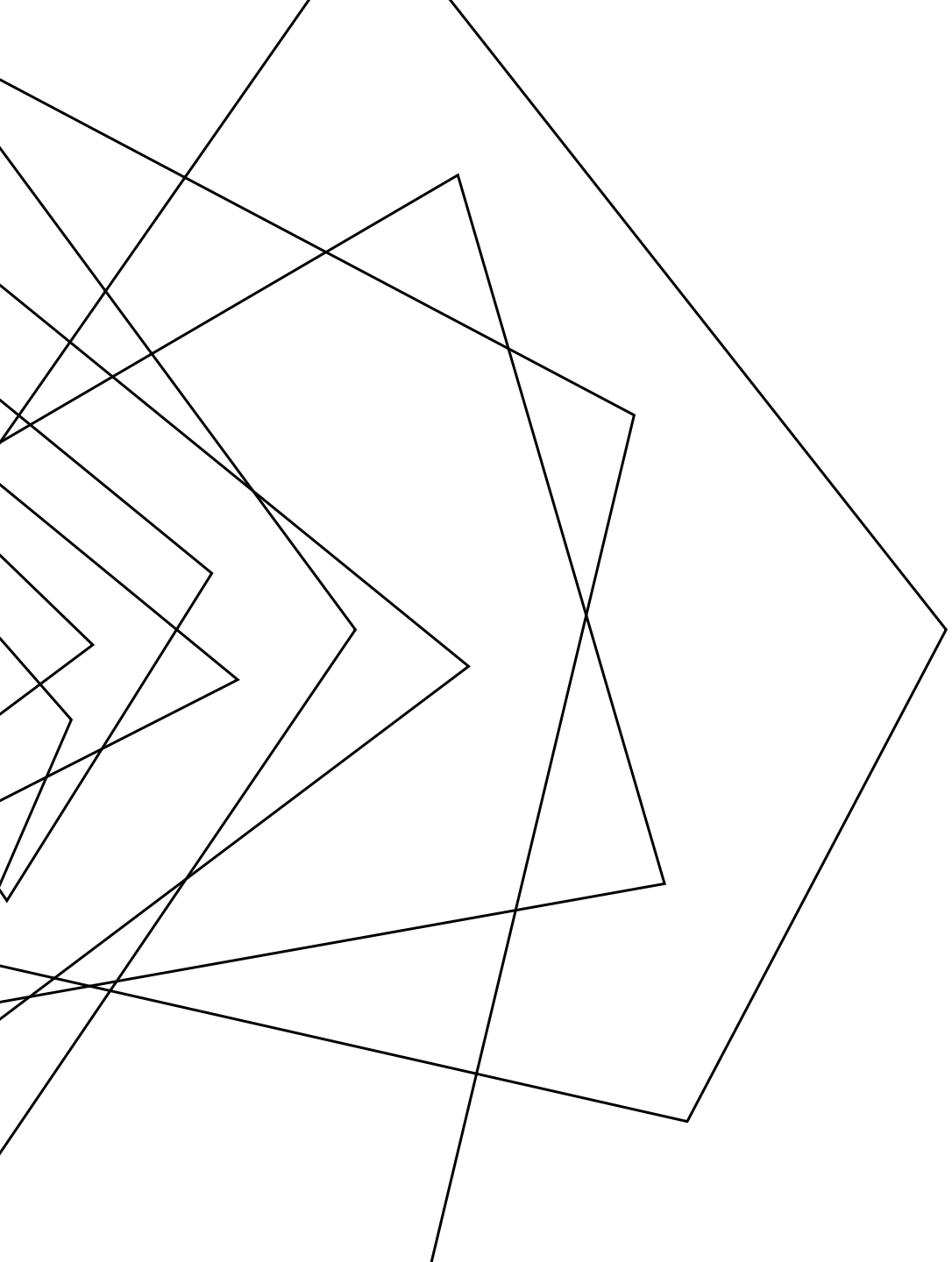
EUF – apply congruence and transitivity to search for contradictions

Lots of clever techniques / optimizations in each theory

Beyond the scope of my ambitions for this class

For much more depth: https://www.decision-procedures.org/





TURNING THE PAGE TO A NEW CHAPTER OF THIS CLASS

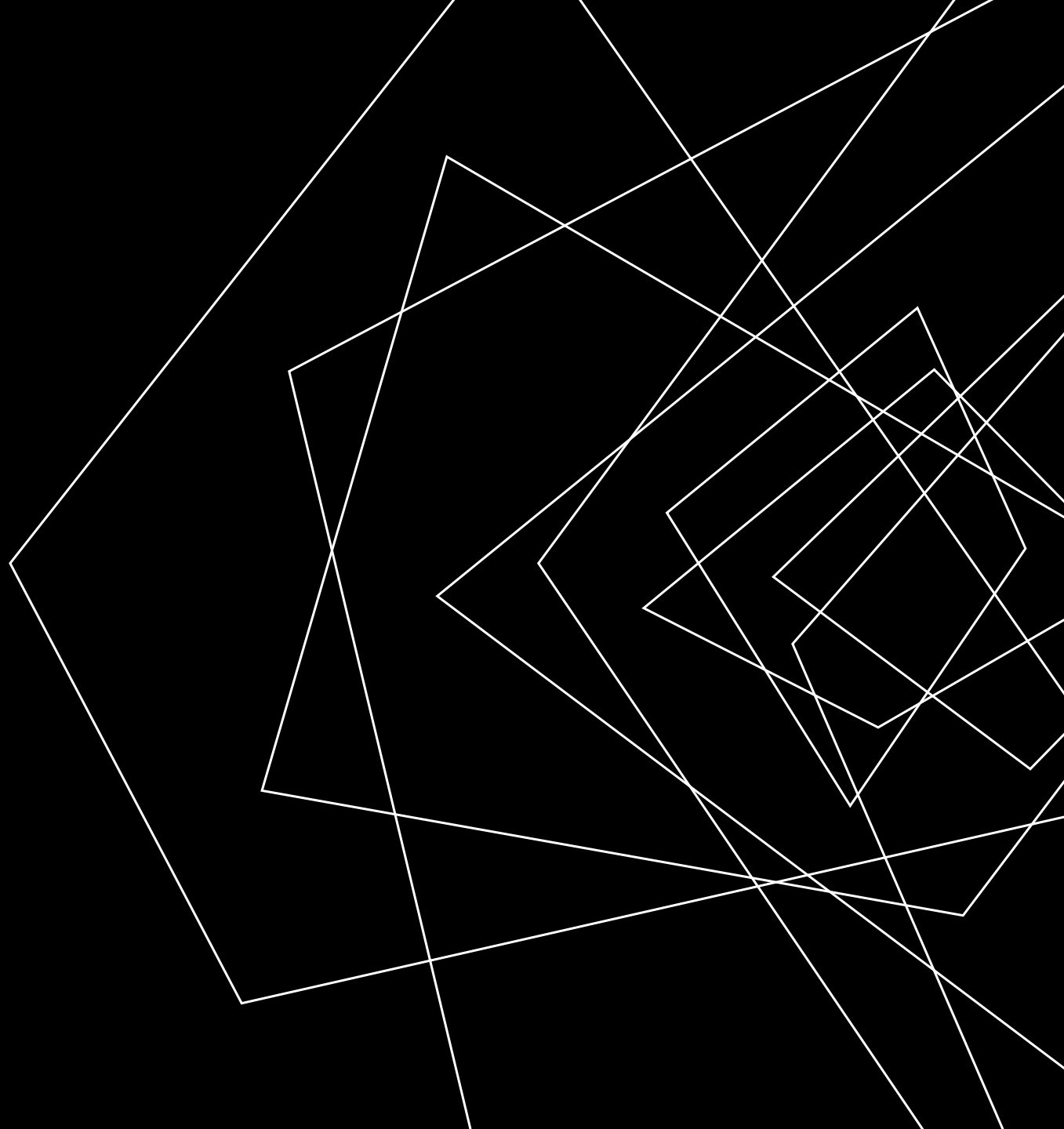
MY GOAL IS TO PROVIDE YOU A SENSE OF THE CHALLENGES AND SOLUTIONS FOR ENGINEERING SECURE SOFTWARE

- TOOLS
- TECHNIQUES
- PROCESSES

Consider the humans

LECTURE OUTLINE

- Human Factors of Security
- Security as Process
- The Secure Software Development Lifecycle



SECURING SOFTWARE IS HARD!

HUMAN FACTORS OF SECURITY

- Surprising Threat Models
- Security-deficient tooling

SOFTWARE: A PATCHWORK OF MANY HANDS

Modern software projects involve teams of programmers

Introducing automated analysis and good development practices may win some battles

Hardening software means developing a commitment and understanding of secure development

Bad news: people are complicated

I hope to convince you to think about the problems that you'll see, place value on security

BOLT-ON SECURITY

LIFECYCLES

Attempting to retrofit a security solution onto a legacy system

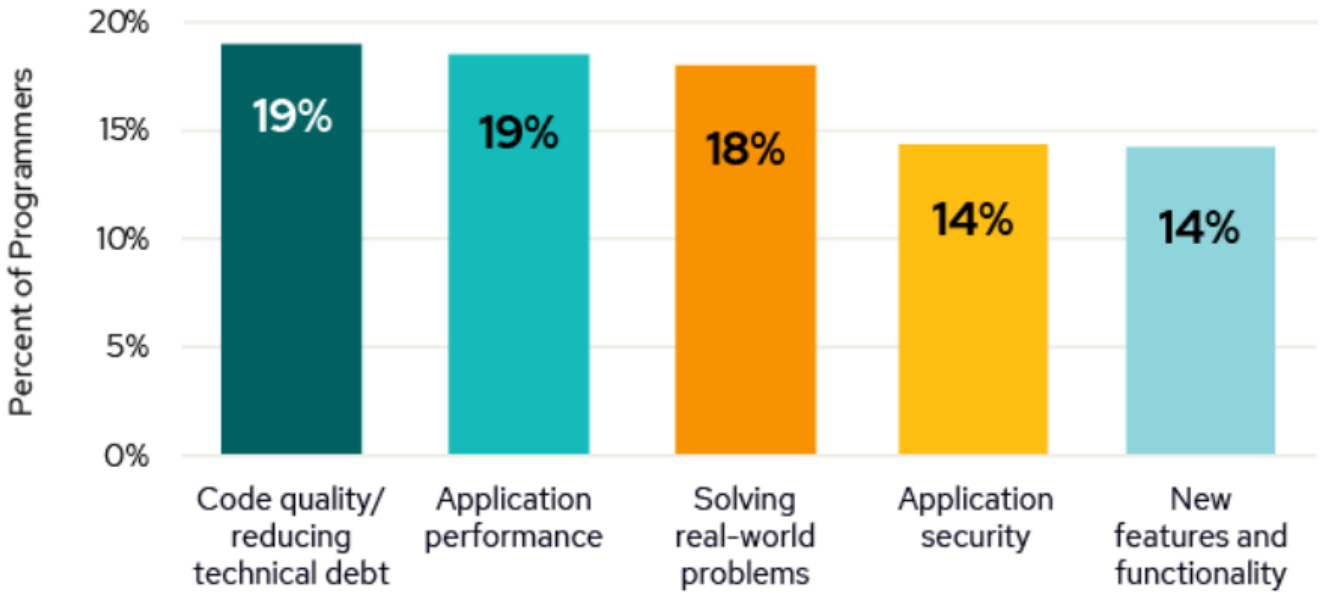
Sometimes necessary

Ideally avoided

VULNERABILITIES IN THE WILD

HUMAN FACTORS OF SECURITY

What is your priority when writing code?

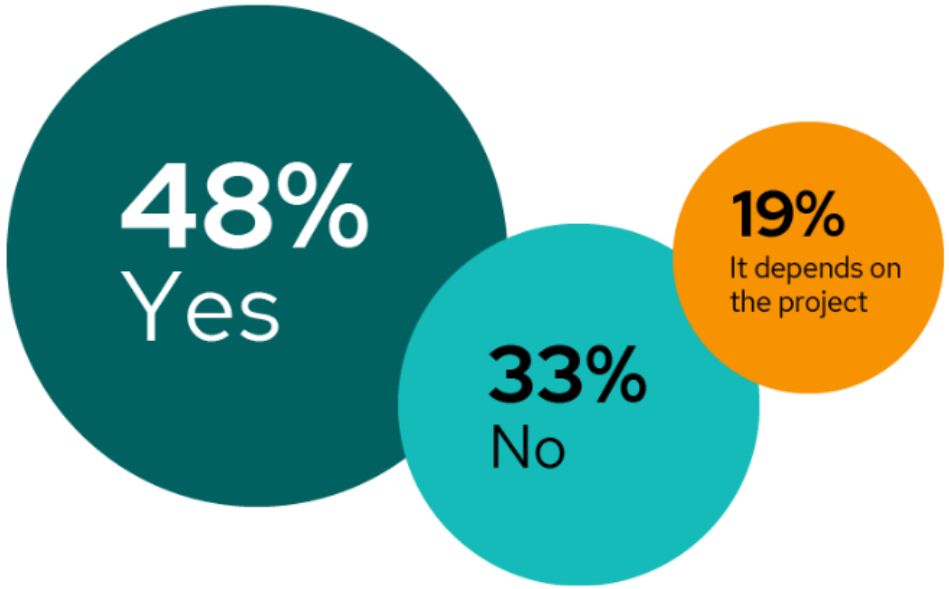


SECURE CODE WARRIOR

VULNERABILITIES IN THE WILD

HUMAN FACTORS OF SECURITY

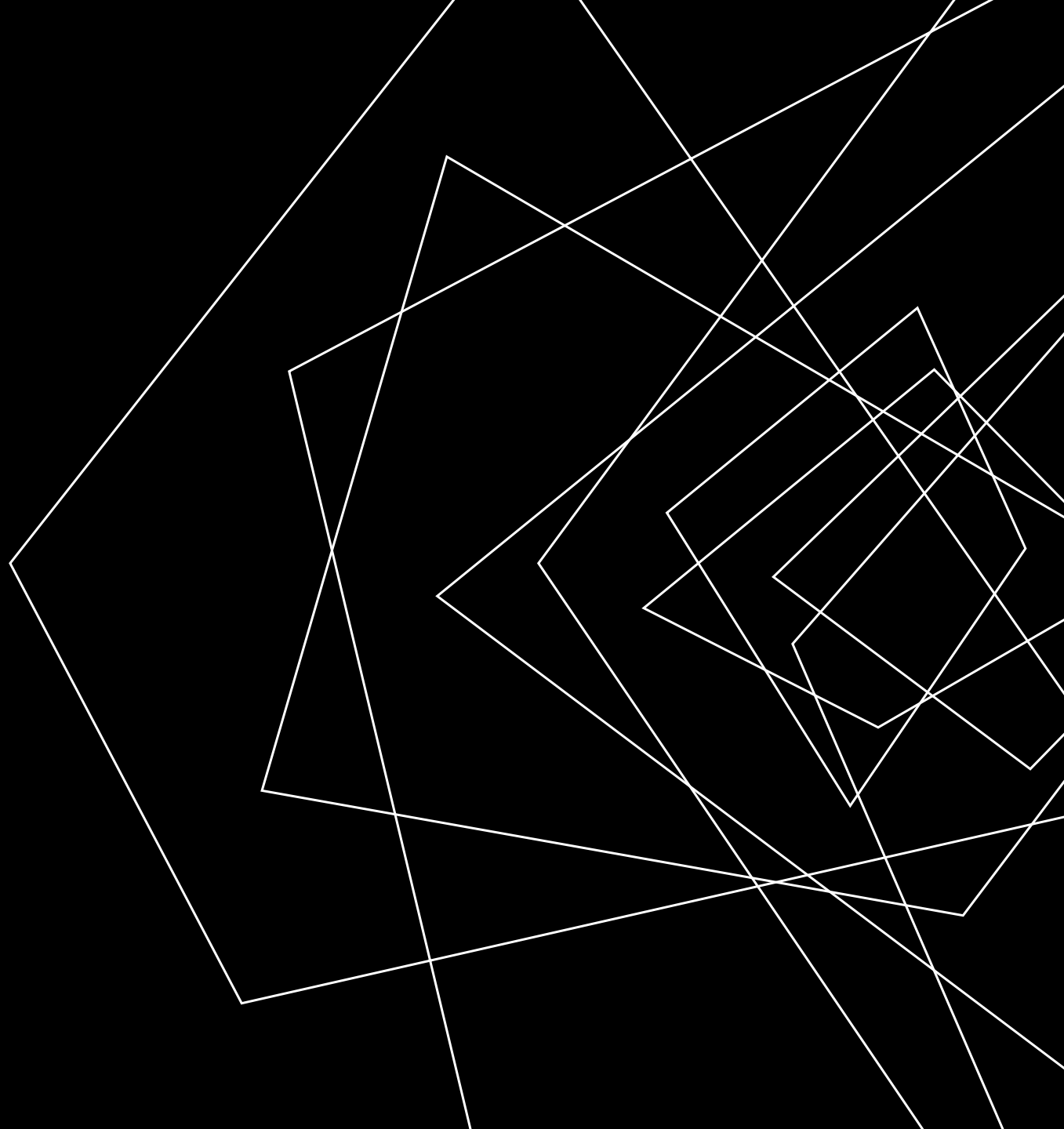
Do you knowingly ship vulnerabilities in code?



SECURE CODE WARRIOR

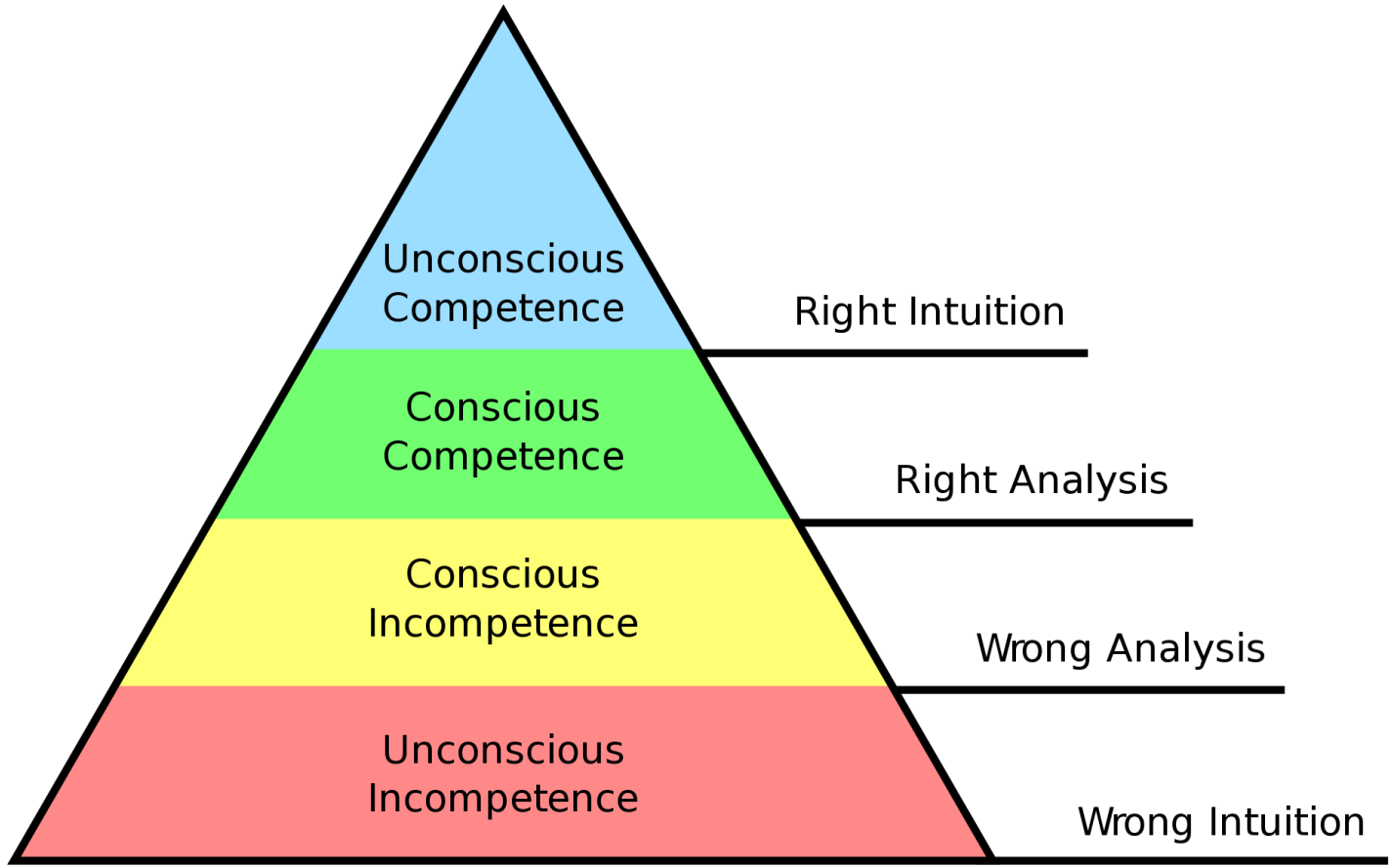
LECTURE OUTLINE

- Human Factors of Security
- Security as Process
- The Secure Software Development Lifecycle



PROCESS IS PROGRESS

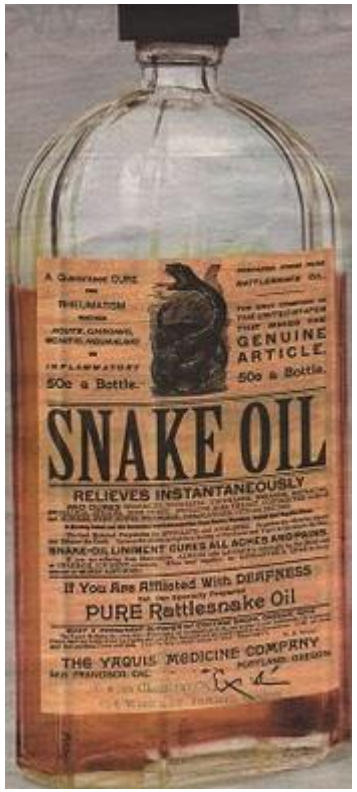
SECURITY AS PROCESS



Hierarchy of Competence

CORPORATE SNAKE OIL

SECURITY AS PROCESS




STORY
TIME

SECURITY VS USABILITY

SECURITY AS PROCESS

A FUNDAMENTAL TENSION

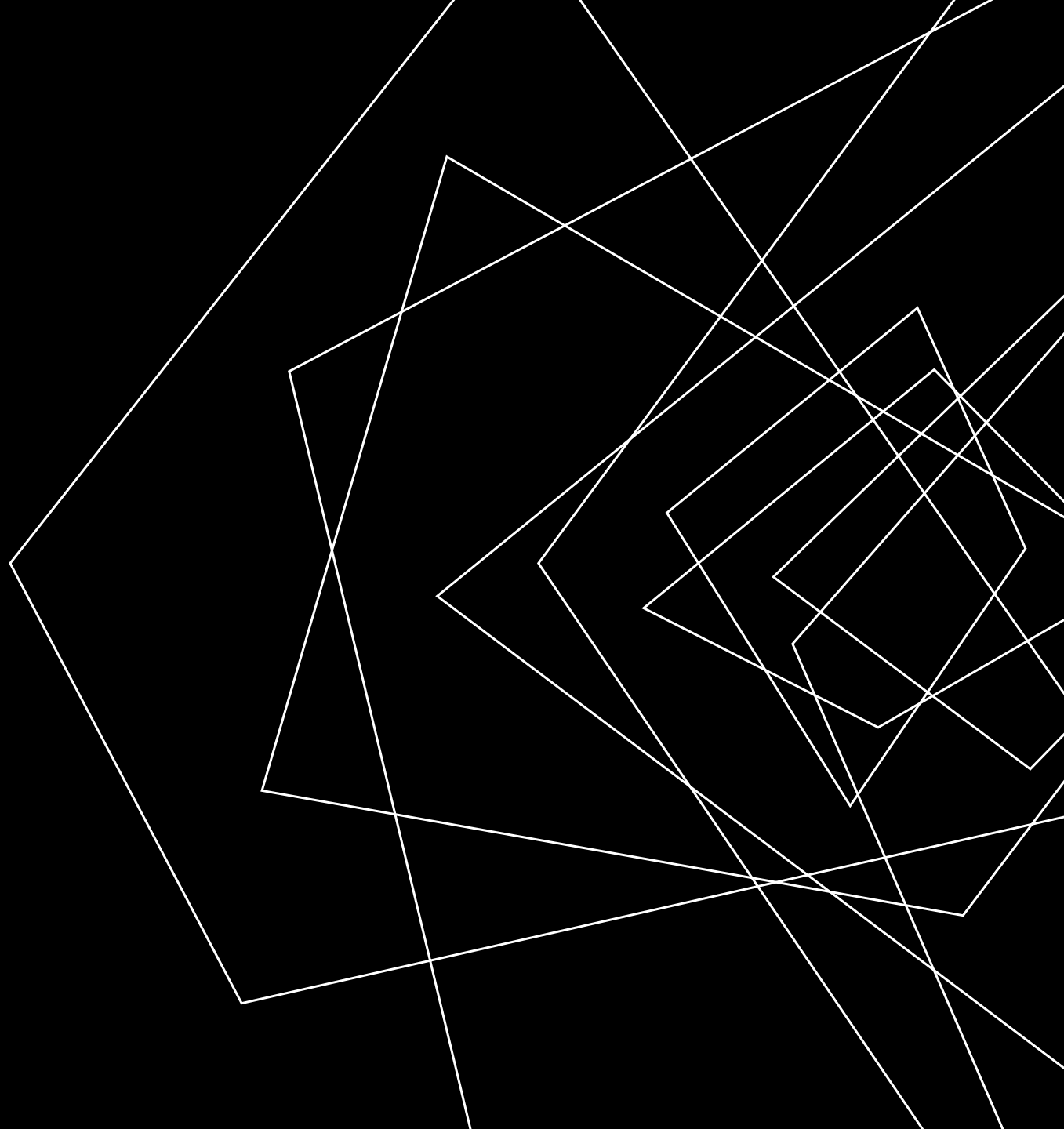
Occurs within the implementation of software, occurs within the processes guiding software development

CONSIDER WHAT WE OWE USERS

Negative externalities

LECTURE OUTLINE

- Human Factors of Security
- Security as Process
- The Secure Software Development Lifecycle



THE “REGULAR” SDLC

LIFECYCLES

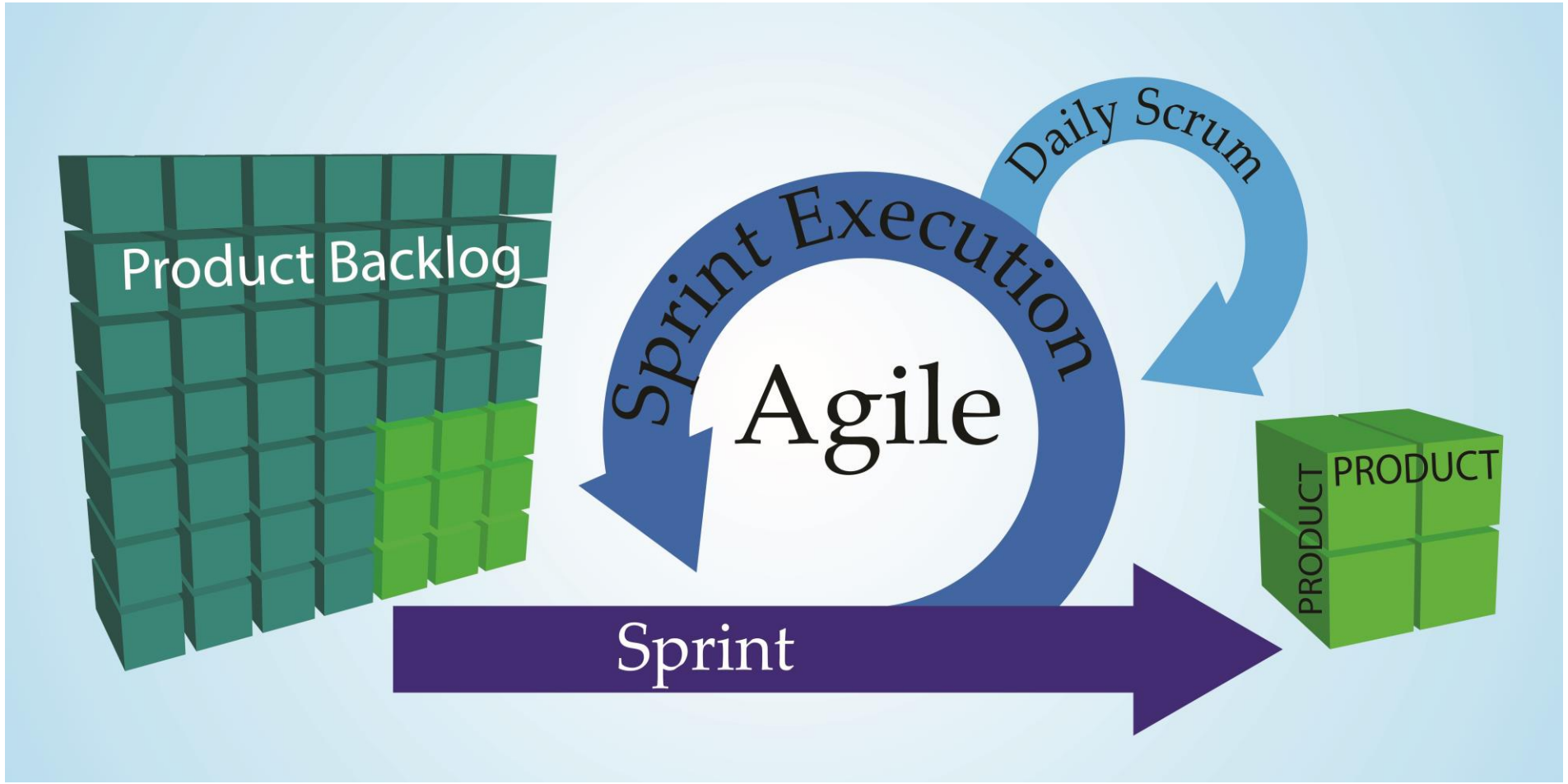
Software Development Life Cycle

- Requirement analysis
- Design
- Development
- Testing and verification
- Deployment
- Maintenance and evolution

The circle of (software) life

AGILE DEVELOPMENT

SDLC: LIFECYCLES



RISK ASSESSMENT AND THREAT MODELS

THE SSDLC COMPONENTS

Companion to Requirement Phase

Functional requirement: User must verify their own contact information

Security consideration: Mechanism misuse

- Users may attempt to access the contact information of others
- Users may attempt to subvert the verification mechanism for harassment

SECURITY DESIGN REVIEW

THE SSDLC COMPONENTS

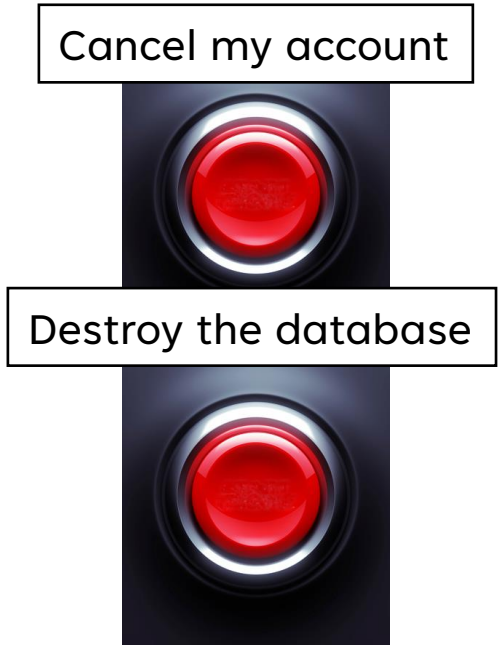
Companion to the Design Phase

Functional requirement: Page should retrieve user's name, email, etc. from customer_info table in database

Security concern: Verify that user has a valid session token before retrieving information from database

Consider Security Design Principles

Principle of least privilege: Do entities in the system have exactly the privileges they need?



Bad design for a button panel

(AUTOMATED) CODE ANALYSIS

THE SSDLC COMPONENTS

COMPANION TO DEVELOPMENT

Apply best practices

- Accessing databases via read-only parameterized queries
- Validating user queries before processing them
- Chaos Engineering

Secure programming

- Assume a function might be misused
- Check arguments for reasonable values
- Canonicalize data

SECURITY TESTING AND CODE REVIEW

THE SSDLC COMPONENTS

COMPANION TO VERIFICATION

Ensure proper use of APIs
- Crypto library invocations
- Holistic audits

Test the test suite:
- Evaluate the coverage of your suite
- Ensure treatment of critical functionality

Value automation:
- Repeatability / reproducibility
- Static analysis!
- Monkey testing

SECURITY ASSESSMENT AND CONFIGURATION

THE SSDLC COMPONENTS

Companion to Maintenance and Evolution

- **Logging** – Capture the behavior of the system (expected AND unexpected)
- **Metrics** – Articulate needs of the system, measure expectations against reality
- **Auditing** – Periodic retrospective analysis over codebase and configuration

WRAP-UP

- Human Factors of Security
- Security as Process
- The Secure Software Development Lifecycle