01. LET'S TALK ABOUT TRUST

CS 361s

Spring 2020

Dr. Seth James Nielson

ABOUT THE INSTRUCTOR

Brigham Young University CS Department Provo, UT

RICE Computer Science

APL JOHNS HOPKINS APPLIED PHYSICS LABORATORY

CRIMSON VISTA TECHNICAL CONSULTING & LITIGATION SUPPORT

WHAT ABOUT YOU?

- Why did you take this course?
- What is your programming background like?
- What has been your favorite course so far? Why?
- What is your learning style?
- What is your favorite teaching style?

PHILOSOPHICAL OVERVIEW

TECHNICAL	NON TECHNICAL
THEORY	PRACTICE
EDUCATED	SKILLED
TEACHER	STUDENT

THE 5 ORDERS OF IGNORANCE

- 0th Order: Known Knowns
- 1st Order: Known Unknowns
- 2nd Order: Unknown Unknowns
- 3rd Order: Unknown methods for discovering unknown unknowns
- 4th Order: Unknown methods for exploring the orders of ignorance

(Adapted from Phillip Armour, "The Five Orders of Ignorance")

THE 5 ORDERS OF IGNORANCE

- 0th Order: Known Knowns
- 1st Order: Known Unknowns

SKILL

- 2nd Order: Unknown Unknowns
- 3rd Order: Unknown methods for discovering unknown unknowns
- 4th Order: Unknown methods for exploring the orders of ignorance

EDUCATION

(Adapted from Phillip Armour, "The Five Orders of Ignorance")

THE WORKLOAD IS CHALLENGING

A **"B"** (or maybe even a "**C**") is not the greatest loss in your academic career. The greatest loss is an "**A**" on your transcript, but a "**B**" (or maybe even a "**C**") in your heart and mind.

Dr Nielson #Quote

A FEW INTRODUCTORY NOTES

- This is my first time teaching this course
- I'm still developing the materials
  - Labwork. Based on previous classes or other curriculum
  - Writing flag. Each lab will follow with a technical writing assignment
- Everything is subject to change. Please be patient.
- Please feel free to make suggestions or raise concerns

CLASS DISCUSSIONS

- I hate slides and I hate "lectures"
- I only use them because I haven't found something better
- What I would like to try is:
  - You read the assignment ahead of time
  - I will prepare slides that can give you some test-prep materials
  - But in-class we will discuss and learn about the topic

THOMPSON'S PAPER

- "Reflections on Trusting Trust"
- 1984.
  - This predates Fred Cohen's seminal work on Viruses
  - This also predates the Morris worm
- What is the foundational concept?

TRUST AND TRUSTWORTHINESS

- I will typically use Ross Anderson's "Security Engineering" for definitions
  - Second Edition (freely available on his website)
  - https://www.cl.cam.ac.uk/~rja14/book.html
- Trusted System – system whose failure can break the security policy
- Trustworthy System – a system that won't break

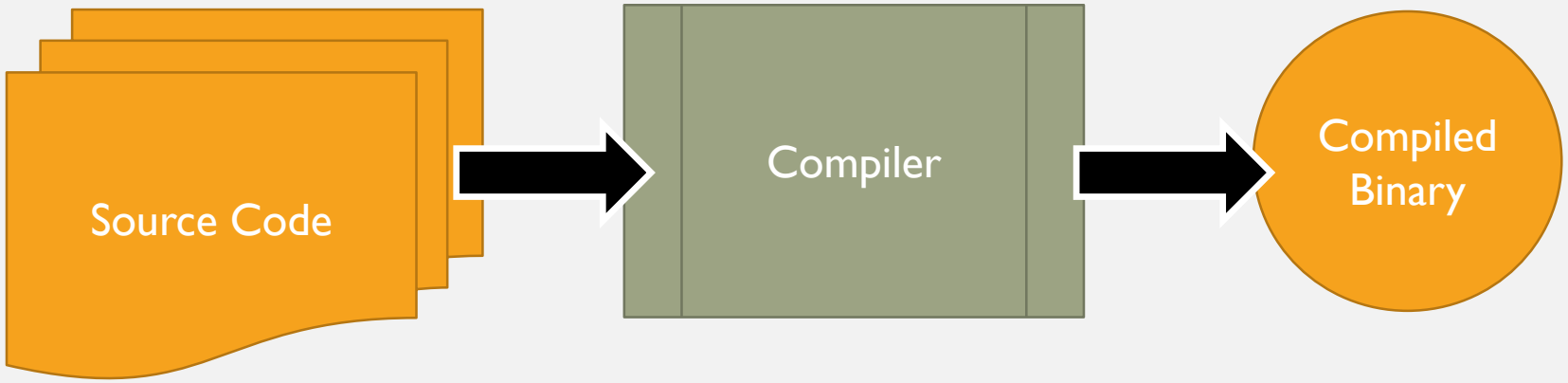
SECURITY IS BASED ON TRUST

- I mean Anderson's concept of trust
- ***We can only enforce security IF the trusted systems don't break***
- How many systems do we need to trust? MANY it turns out.

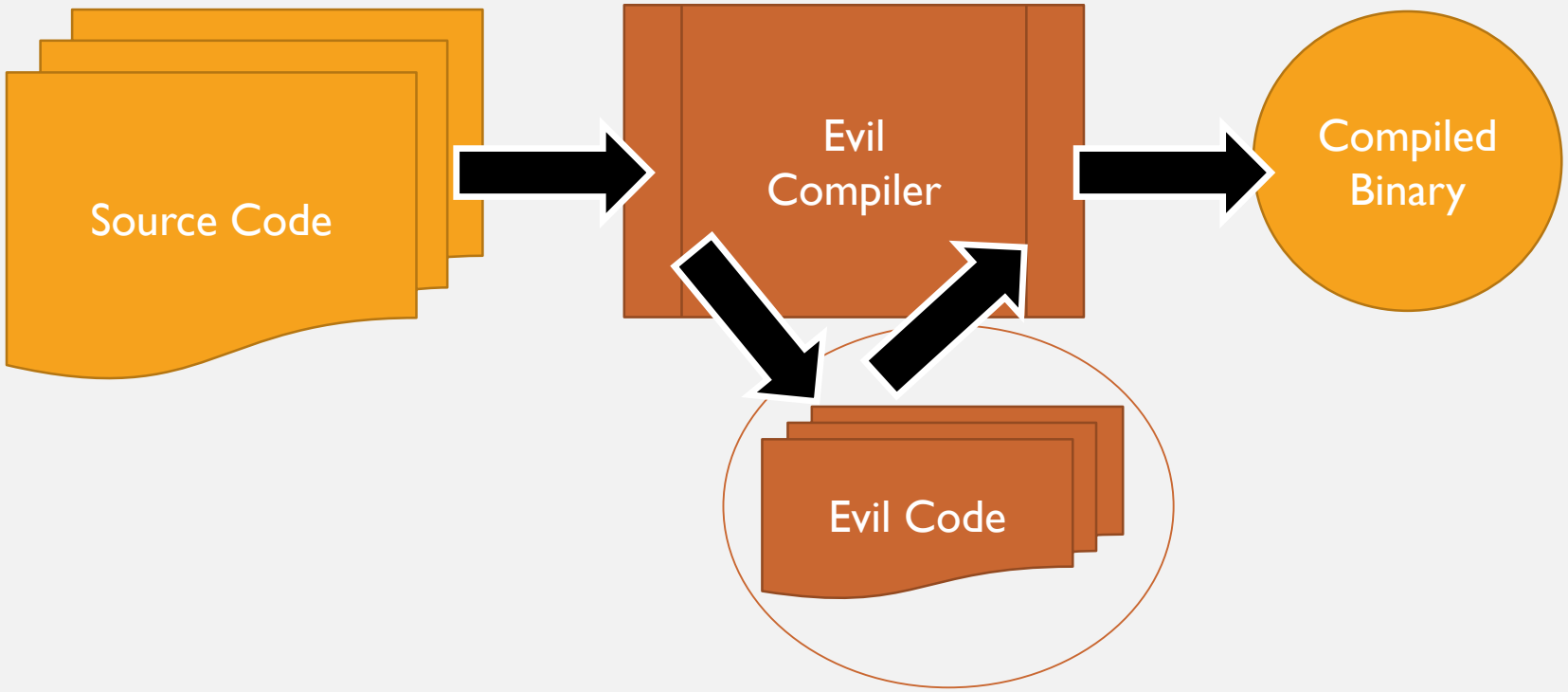
WHAT THOMPSON TAUGHT

- Your program isn't trustworthy if the compiler isn't trustworthy
- The compiler isn't trustworthy if it's compiler isn't trustworthy
- The compiler's compiler isn't trustworthy if it's compiler isn't trustworthy
- … on forever back to the first compiler built by assembly

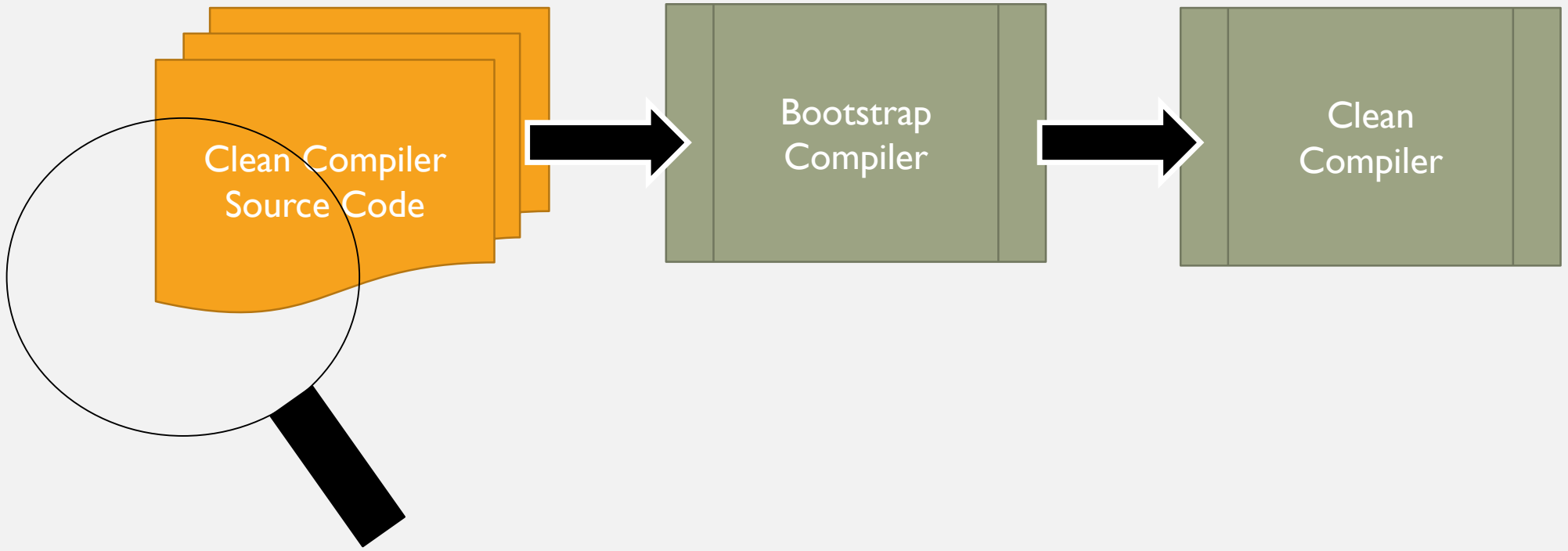
THE NORMAL COMPILATION PROCESS



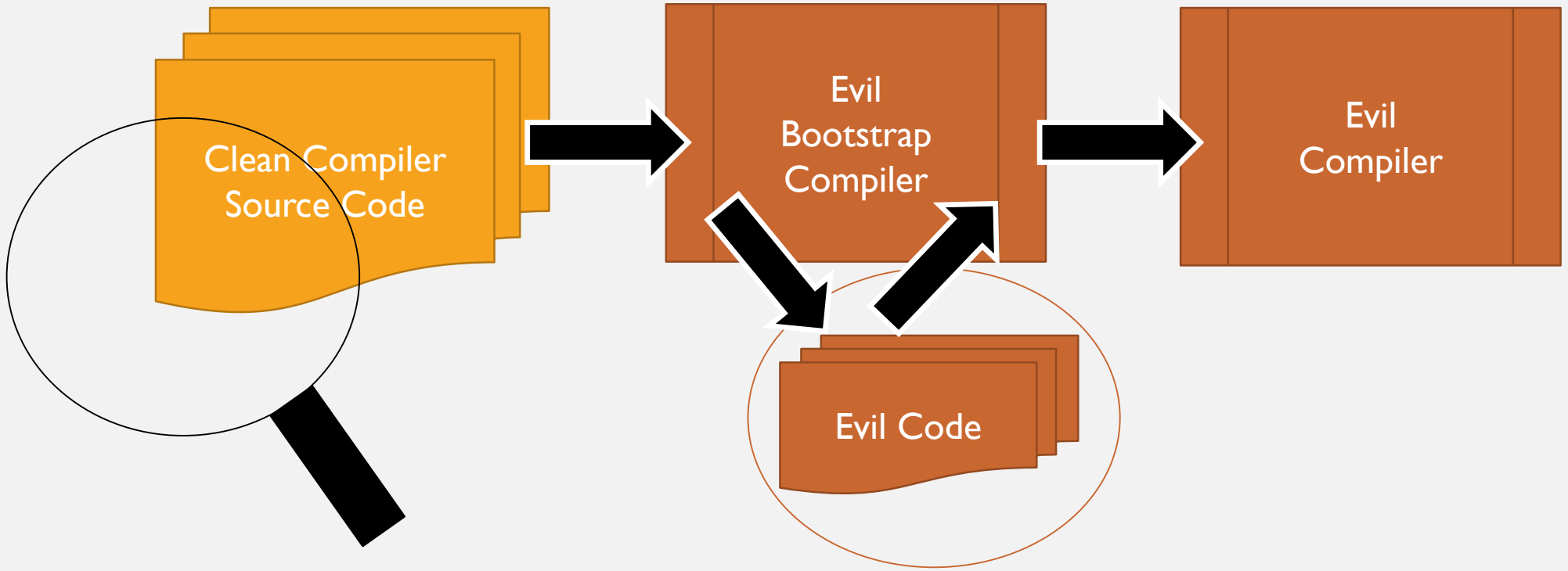
EVIL COMPILATION



INSPECT THE COMPILER SOURCE?



NOT GOOD ENOUGH!



WHAT IS SECURITY REALLY?

- What does it mean to say a system is "secure"?
- Anderson:
  - Security Policy – What is protected
  - Mechanism – How it is protected
  - Assurance – How confident are we that the mechanism enforces policy
- A "secure" system is more-or-less one with a lot of assurance

HOW DO WE GET ASSURANCE?

- Testing
- Formal Proofs
- Cryptographic Proofs
- Metrics, History, Experience

TRUST CHAINS

- Much of our assurance ***assumes*** axiomatic trustworthiness elsewhere
  - Compilers
  - ***The compilers that built the compilers!***
  - Third party libraries
  - Operating systems ***including device drivers***
  - Hardware
  - Root certificates
  - Trusted Third Parties

IN THIS CLASS

- You will show a lot of things we trust are not trustworthy
  - Lab 1 – corrupt compilers
  - Lab 2 and 3 – break programs using buffer overflows and ROP
  - Lab 4 – break web APIs
  - Lab 5 – break encryption/authentication using a corrupted root certificate
  - Lab 6 – break sandboxes/virtual machines/emulators