The Plan

- Administrivia
- Bug hunting strategy
- In-class build/break time!
• Semi-deadline extension

• Source for everyone’s project is in the break_source repo

• Teams that did not finish have an additional day extension

• If your code works locally, but not on the grading server, please show us and we will give you points for it

• Design doc v2 and mid-course surveys due tomorrow (1/16) by midnight

There will not be another extension
Administrivia

Break updates

• No correctness bugs
• Only 3 breaks required for min criterion
• Submit break descriptions for teams whose code can’t compare to the oracle
  • Include json test case if it works locally
  • State in that it is infeasible to break because their code doesn’t run
• Worth 100 points, but does not accumulate over time
How to find bugs?
Bug Hunters!

Adversarial Mindset!

General bug finders:
• Functionality
• Performance
• Security

White-hat Hackers:
• Security Team
• Penetration Testers
• Bug Bounty
Questions

1. How do Testers and Hackers search for vulnerabilities?

2. What is the difference between the groups?

**Interview and Observations:**

- Vulnerability Discovery task analysis
- Reverse engineering observations
Questions

1. How do Testers and Hackers search for vulnerabilities?

2. What is the difference between the groups?
Info Gathering

Program Understanding

Attack Surface

Exploration

Vulnerability Recognition

Reporting
• Build context prior to reading or executing code

• Example actions:
  • Language
  • Libraries
  • File sizes
  • Bug history
• Determine how the program operates
  • Interaction between components
  • Interaction with the environment

• Example actions:
  • Run the code with basic inputs
  • Scan for important functions by name

“You’re touching a little bit everything, and then you are organizing that into a structure in your head.”
• Identify how user interacts with program

• Direct and indirect inputs

• Example actions:
  • Identify code handling inputs
    • Command line
    • Programs from the network
    • Config file

“If we allow logins from a 3rd party provider, I would check the way we handle that data.”
• **Possible inputs to the attack surface**

• **Example actions:**
  • Fuzzing (manual/automated)
  • Read code
    • Skim control/data flow paths
• Notice a problem when exploring
• Typically **intuition**-based

“I have in my mind a set of possible bugs…if there’s a loop going through my input, it could be going out of bound on the array…and I start to look at the code. See if something comes to my mind.”

“Basically there's a library of considerations that you develop through experience and understanding the code base and the products.”
• Tell developers about the problem
• Advocate for remediation
• Critical aspects:
  • Make report understandable
  • Importance of fixing

“You do have to [convince] someone that there’s a risk. …It’s quite timely [time consuming], running a ticket.”
Questions

1. How do Testers and Hackers search for vulnerabilities?

2. What is the difference between the groups?
Employment

Hacking Exercises

Community

Bug Reports

Vulnerability Discovery Experience

Attack Surface

Exploration

Vulnerability Recognition
Summary

• Bug hunting strategy
  • Information Gathering
  • Program Understanding
  • Attack Surface
  • Exploration
  • Vulnerability Recognition
  • Reporting

• Vulnerability discovery skill development
In-class Build/Break Time!

• Divide up into teams and spread out
  • You can leave this room, but stay on this floor
  • Send us a message in Slack with where you go

• Things to do:
  • Finish build
  • Design doc v2
  • Strategize for break round