CMPT 479/886
Automated Software Analysis & Security
Nick Sumner

Much adapted from Xiangyu Zhang, Antony Hosking, Sorin Lerner, Jonathan Aldrich, Sam Blackshear
Course Website

- www.cs.sfu.ca/~wsumner/teaching/886/17f/
  - Schedule
  - Policies
  - Assignments
  - Paper Suggestions
Why are you here?

- Programs are big, complex, and difficult to reason about.
Why are you here?

- Programs are big, complex, and difficult to reason about.

1001101
0101011
1101011
1101011
0001110
Why are you here?

- Programs are big, complex, and difficult to reason about.
Why are you here?

- Programs are big, complex, and difficult to reason about.

```
1001101
0101011
1101011
0001110
```

```
 Foo.c
 Bar.c
 Baz.c
```
Why are you here?

• Programs are big, complex, and difficult to reason about.

1001101
0101011
1101011
0001110

Foo.c
Bar.c
Baz.c
Why are you here?

- Programs are big, complex, and difficult to reason about.
Why are you here?

- Programs are big, complex, and difficult to reason about.

1001101 0101011 1101011 0001110

Foo.c
Bar.c
Baz.c
Why are you here?

• Programs are big, complex, and difficult to reason about.

1001101
0101011
1101011
0001110

Foo.c
Bar.c
Baz.c

A+
Why are you here?

• Programs are big, complex, and difficult to reason about.

```
1001101
0101011
1101011
0001110
```

```
Foo.c
Bar.c
Baz.c
```
Why are you here?

- Programs are big, complex, and difficult to reason about.
Why are you here?

- Programs are big, complex, and difficult to reason about.
Why are you here?

- Programs are big, complex, and difficult to reason about.

Are there more efficient designs?
Why are you here?

- Programs are big, complex, and difficult to reason about.

  Are there more efficient designs?  
  What is the cause of a bug?
Why are you here?

- Programs are big, complex, and difficult to reason about.

  Are there more efficient designs?

  What is the cause of a bug?

  How do I find new bugs?
Why are you here?

- Programs are big, complex, and difficult to reason about.

- Are there more efficient designs?
- What is the cause of a bug?
- How do I find new bugs?
- How do I find security vulnerabilities?
Why are you here?

- Programs are big, complex, and difficult to reason about.

Are there more efficient designs?

What is the cause of a bug?

How do I find new bugs?

How do I find security vulnerabilities?

Can I protect against them?
Why are you here?

- Programs are big, complex, and difficult to reason about.

People are bad at tedious, subtle tasks, but computers are great at them!
Goal

- Learn how the difficult tasks in development can be pushed onto computers.
Goal

- Learn how the difficult tasks in development can be pushed onto computers.
  - Survey of *program analysis* techniques & papers
Goal

• Learn how the difficult tasks in development can be pushed onto computers.
  – Survey of program analysis techniques & papers
    • Profiling
      (Speed, Potential Concurrency, Memory, ...)
Goal

• Learn how the difficult tasks in development can be pushed onto computers.
  – Survey of program analysis techniques & papers
    • Profiling
    • Testing

More effective tests. Bridge testing & verification
Goal

• Learn how the difficult tasks in development can be pushed onto computers.
  – Survey of *program analysis* techniques & papers
    • Profiling
    • Testing
    • Debugging

  Explaining or locating the causes of bugs
Goal

- Learn how the difficult tasks in development can be pushed onto computers.
  - Survey of program analysis techniques & papers
    - Profiling
    - Testing
    - Debugging
    - Concurrency

How to explain race conditions?
Atomicity violations?
How to find 'Heisenbugs'?
Goal

- Learn how the difficult tasks in development can be pushed onto computers.
  - Survey of *program analysis* techniques & papers
    - Profiling
    - Testing
    - Debugging
    - Concurrency
    - Security

How to find vulnerabilities before attackers.
(...or as attackers)
Goal

- Learn how the difficult tasks in development can be pushed onto computers.
  - Survey of *program analysis* techniques & papers
    - Profiling
    - Testing
    - Debugging
    - Concurrency
    - Security
    - Verification

How to prove the absence of behaviors.
Guiding questions:
Guiding questions:

- These problems are *impossible* to precisely solve in general.
Guiding questions:

- These problems are *impossible* to precisely solve in general. *What are the compromises?*
Guiding questions:

- These problems are *impossible* to precisely solve in general. What are the compromises?
  - What cornercases make them fail?
Lens

Guiding questions:

• These problems are *impossible* to precisely solve in general. What are the compromises?
  – What corner cases make them fail?
  – *Why* do these corner cases exist?
Lens

Guiding questions:

• These problems are *impossible* to precisely solve in general. What are the compromises?
  – What cornercases make them fail?
  – Why do these cornercases exist?

• How do authors present their work? Why?
Guiding questions:

- These problems are *impossible* to precisely solve in general. What are the compromises?
  - What cornercases make them fail?
  - Why do these cornercases exist?

- How do authors present their work? Why?
  - What is highlighted? What is hidden?
Lens

Guiding questions:

- These problems are *impossible* to precisely solve in general. What are the compromises?
  - What cornercases make them fail?
  - Why do these cornercases exist?

- How do authors present their work? Why?
  - What is highlighted? What is hidden?
  - How is it evaluated?
Structure

- First few weeks are review & background
  - I present.
  - There may be quizzes
  - You think about papers you'd like to present
Structure

• First few weeks are review & background
  – I present.
  – There may be quizzes
  – You think about papers you'd like to present

• Reading foundational & new papers
  – 2 student presentations & paper discussions per week
  – Brief critique (1-2 pages) on weeks you don't present
Structure

• First few weeks are review & background
  – I present.
  – There may be quizzes
  – You think about papers you'd like to present

• Reading foundational & new papers
  – 2 student presentations & paper discussions per week
  – Brief critique on weeks you don't present

• 4 small projects to introduce core skills
Structure

• First few weeks are review & background
  – I present.
  – There may be quizzes
  – You think about papers you'd like to present

• Reading foundational & new papers
  – 2 student presentations & paper discussions per week
  – Brief critique on weeks you don't present

• 4 small projects to introduce core skills

Available now!
Structure

• First few weeks are review & background
  – I present.
  – There may be quizzes
  – You think about papers you'd like to present
• Reading foundational & new papers
  – 2 student presentations & paper discussions per week
  – Brief critique on weeks you don't present
• 4 small projects to introduce core skills
• 1 large course project
Presentations

• Guidelines on website

• 2 Goals
  – Help reinforce the material for the class
  – Lead an interesting discussion to examine the trade offs of each technique. (I'll be helping.)
Presentations

• Guidelines on website

• 2 Goals
  – Help reinforce the material for the class
  – Lead an interesting discussion to examine the trade offs of each technique. (I'll be helping.)

• Show how the technique behaves in the best case

• Show or lead discussion on where it might behave poorly
Presentations

• Guidelines on website

• 2 Goals
  – Help reinforce the material for the class
  – Lead an interesting discussion to examine the trade offs of each technique. (I'll be helping.)

• Show how the technique behaves in the best case

• Show or lead discussion on where it might behave poorly

• Groups of TBD (5?) will present each paper.
  – Volunteer or be volunteered
Critiques

- Guidelines on website
- 1-2 page response to 1 paper each week that you do not present.
- Primarily meant to prepare you for the discussion on the paper that week.
Term Projects

- Groups of 4. (Grad groups can be smaller)
- 1 page proposals due October 10.
- Brief meetings with me on October 11.
Term Projects

- Groups of 4. (Grad groups can be smaller)
- 1 page proposals due October 10.
- Brief meeting with me on October 11.
- Find something that interests (or irritates) you and go after it!
  - Maybe look at how these techniques can help your existing research
Participation

- A class of this nature is driven by *discussion*.
Participation

- A class of this nature is driven by *discussion*.
  - You should not just show up but also contribute.
Participation

• A class of this nature is driven by *discussion*.
  – You should not just show up but also contribute.
  – Even the projects may require discussion for you to succeed.
Participation

• A class of this nature is driven by *discussion*.
  – You should not just show up but also contribute.
  – Even the projects may require discussion for you to succeed.

• Think about things in advance.
Participation

- A class of this nature is driven by *discussion*.
  - You should not just show up but also contribute.
  - Even the projects may require discussion for you to succeed.
- Think about things in advance.
- Come to class with questions (or answers).
What Could We Look At?

- Surviving Failures
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
- Automated Test Generation
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
- Automated Test Generation
- Automated Regression Testing
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
- Automated Test Generation
- Automated Regression Testing
- Program Guided Fuzz Testing
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
- Automated Test Generation
- Automated Regression Testing
- Program Guided Fuzz Testing
- Data Race Explanation
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
- Automated Test Generation
- Automated Regression Testing
- Program Guided Fuzz Testing
- Data Race Explanation
- Battery Use Profiling
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
- Automated Test Generation
- Automated Regression Testing
- Program Guided Fuzz Testing
- Data Race Explanation

- Battery Use Profiling
- Mobile Privilege Protection/Reduction
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
- Automated Test Generation
- Automated Regression Testing
- Program Guided Fuzz Testing
- Data Race Explanation
- Battery Use Profiling
- Mobile Privilege Protection/Reduction
- Reproducing Remote Bugs
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
- Automated Test Generation
- Automated Regression Testing
- Program Guided Fuzz Testing
- Data Race Explanation

- Battery Use Profiling
- Mobile Privilege Protection/Reduction
- Reproducing Remote Bugs
- Program Verification
What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
- Automated Test Generation
- Automated Regression Testing
- Program Guided Fuzz Testing
- Data Race Explanation

- Battery Use Profiling
- Mobile Privilege Protection/Reduction
- Reproducing Remote Bugs
- Program Verification
- Automated Program Repair
- ...


What Could We Look At?

- Surviving Failures
- Plagiarism Detection
- Malware Detection
- Identifying Information Leaks
- Automated Debugging
- Automated Test Generation
- Automated Regression Testing
- Program Guided Fuzz Testing
- Data Race Explanation
- Battery Use Profiling
- Mobile Privilege Protection/Reduction
- Reproducing Remote Bugs
- Program Verification
- Automated Program Repair
-...

I have planned out a survey, but we can customize it for interest