CSCI 2951U: Topics in Software Security

Introduction

Vasileios (Vasilis) Kemerlis

January 25, 2021

Department of Computer Science
Brown University
What is this course about?

State-of-the-art in software exploitation and defense

CSCI 1650++

Memory unsafe code (written in C/C++, asm, ...)

1. Prevalent software defects
   - Stack/Heap smashing
   - Format string bugs
   - Pointer errors
   - ...

2. Modern defenses
   - W^X, ASLR
   - Stack/Heap canaries
   - RELRO, BIND NOW
   - BPF_SECCOMP, FORTIFY_SRC
   - CFI, CPI, ...

Software Exploitation

1. Code injection
2. Code reuse
   - Return-to-libc (ret2libc)
   - Return-oriented prog. (ROP)
   - Just-In-Time ROP (JIT-ROP)
   - Blind ROP (BROP)
   - Signal-oriented prog. (SROP)
   - ...

3. Data-only attacks

vpk@cs.brown.edu (Brown University)
Course Overview (1/2)

▶ What is this course about?

✔ State-of-the-art in software exploitation and defense ➔ CSCI 1650++

Memory unsafe code (written in C/C++, asm, ...)

Software Security
1. Prevalent software defects
   • Stack/Heap smashing
   • Format string bugs
   • Pointer errors
   • ...
2. Modern defenses
   • W^X, ASLR
   • Stack/Heap canaries
   • RELRO, BIND_NOW
   • BPF_SECCOMP, FORTIFY_SRC
   • CFI, CPI, ...

Software Exploitation
1. Code injection
2. Code reuse
   • Return-to-libc (ret2libc)
   • Return-oriented prog. (ROP)
   • Just-In-Time ROP (JIT-ROP)
   • Blind ROP (BROP)
   • Signal-oriented prog. (SROP)
   • ...
3. Data-only attacks
What is this course about?

- State-of-the-art in software exploitation and defense ➔ CSCI 1650++
- Memory unsafe code (written in C/C++, asm, ...)

Software Security

1. Prevalent software defects
   - Stack/Heap smashing
   - Format string bugs
   - Pointer errors
   - ...

2. Modern defenses
   - W^X, ASLR
   - Stack/Heap canaries
   - RELRO, BIND_NOW
   - BPF_SECCOMP, FORTIFY_SRC
   - CFI, CPI, ...

Software Exploitation

1. Code injection
2. Code reuse
   - Return-to-libc (ret2libc)
   - Return-oriented prog. (ROP)
   - Just-In-Time ROP (JIT-ROP)
   - Blind ROP (BROP)
   - Signal-oriented prog. (SROP)
   - ...
3. Data-only attacks
▶ Why take this course?

- Understand the boundaries of protection mechanisms and argue about their effectiveness
- Familiarize with experimental mitigation techniques
- Learn how and why (certain) defenses can be bypassed

Why are these useful?

- To design effective (and efficient) software protection mechanisms you need to:
  (a) understand what sorts of attacks are possible
  (b) how exactly these attacks work
  (c) why previous attempts failed
Why take this course?

- **Offense**
  - Learn how and why (certain) defenses can be **bypassed**
    - Exploit “weaponization”
### Why take this course?

<table>
<thead>
<tr>
<th>Defense</th>
<th>Offense</th>
</tr>
</thead>
</table>
| ✓ Understand the **boundaries** of protection mechanisms and argue about their **effectiveness** | ✓ Learn **how** and **why** (certain) defenses can be **bypassed**  
  - Exploit “weaponization” |
| ✓ Familiarize with experimental mitigation techniques |   |
Why take this course?

**Defense**
- Understand the **boundaries** of protection mechanisms and argue about their **effectiveness**
- **Familiarize** with experimental mitigation techniques

**Offense**
- Learn **how** and **why** (certain) defenses can be **bypassed**
  - Exploit “weaponization”

Why are these useful?

- To design effective (and efficient) software protection mechanisms you need to:
  - understand **what** sorts of attacks are possible
  - **how** exactly these attacks work
  - **why** previous attempts failed
Prerequisites

- **CSCI 1650 (Software Security and Exploitation)**
  - Control-flow Hijacking
  - Code Injection (Shellcode dev.)
  - Code Reuse (ROP)

- **CSCI 1670 (Operating Systems)**
  - C/C++, x86 \texttt{asm}
  - Linking and Loading
  - Virtual Memory

Having taken the following courses is a plus, but not required:
- **CSCI 1660 (Computer Systems Security)**
- **CSCI 2951E (Topics in Computer System Security)**

We will review (most of) the important concepts.
Prerequisites

- **CSCI 1650 (Software Security and Exploitation)**
  - Control-flow Hijacking
  - Code Injection (Shellcode dev.)
  - Code Reuse (ROP)

- **CSCI 1670 (Operating Systems)**
  - C/C++, x86 *asm*
  - Linking and Loading
  - Virtual Memory

- ✔ Having taken the following courses is a plus, but not required:
  - **CSCI 1660 (Computer Systems Security)**
  - **CSCI 2951E (Topics in Computer System Security)**
Prerequisites

- **CSCI 1650** *(Software Security and Exploitation)*
  - Control-flow Hijacking
  - Code Injection (Shellcode dev.)
  - Code Reuse (ROP)

- **CSCI 1670** *(Operating Systems)*
  - C/C++, x86 *asm*
  - Linking and Loading
  - Virtual Memory

- Having taken the following courses is a plus, but not required:
  - **CSCI 1660** *(Computer Systems Security)*
  - **CSCI 2951E** *(Topics in Computer System Security)*

- We will review (most of) the important concepts
Logistics

Meetings
• Mondays, 3PM – 5:20PM (M hour)

Grading
- Paper reviews: 10%
- Paper presentations: 20%
- Discussion part: 20%
- Project report: 40%
- Project presentation: 10%

Communication
- https://cs.brown.edu/courses/csci2951-u/
- course.csci.2951u.2021-spring.s01@lists.brown.edu
- Check the website!
- Announcements
- Lecture slides
- Readings

Study material
- No required textbook
- Assigned readings

vpk@cs.brown.edu (Brown University) CSCI 2951U Spring ’21
Logistics

 Meetings

- Mondays, 3PM – 5:20PM (*M hour*)
- Zoom
## Logistics

<table>
<thead>
<tr>
<th>Meetings</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Mondays, 3PM – 5:20PM (<em>M hour</em>)</td>
</tr>
<tr>
<td>- Zoom</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Communication</th>
</tr>
</thead>
<tbody>
<tr>
<td>- <a href="https://cs.brown.edu/courses/csci2951-u/">https://cs.brown.edu/courses/csci2951-u/</a></td>
</tr>
<tr>
<td>- <a href="mailto:course.csci.2951u.2021-spring.s01@lists.brown.edu">course.csci.2951u.2021-spring.s01@lists.brown.edu</a></td>
</tr>
</tbody>
</table>

vpk@cs.brown.edu (Brown University)
Logistics

Meetings

• Mondays, 3PM – 5:20PM (M hour)
• Zoom

Communication

• https://cs.brown.edu/courses/csci2951-u/
• course.csci.2951u.2021-spring.s01@lists.brown.edu

Check the website!

• Announcements
• Lecture slides
• Readings
### Logistics

**Meetings**
- Mondays, 3PM – 5:20PM *(M hour)*
- Zoom

<table>
<thead>
<tr>
<th>Grading</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️ Paper reviews ➔ 10%</td>
</tr>
<tr>
<td>✔️ Paper presentations ➔ 20%</td>
</tr>
<tr>
<td>✔️ Discussion part. ➔ 20%</td>
</tr>
<tr>
<td>✔️ Project report ➔ 40%</td>
</tr>
<tr>
<td>✔️ Project presentation ➔ 10%</td>
</tr>
</tbody>
</table>

**Communication**
- [https://cs.brown.edu/courses/csci2951-u/](https://cs.brown.edu/courses/csci2951-u/)
- course.csci.2951u.2021-spring.s01@lists.brown.edu

**Check the website!**
- Announcements
- Lecture slides
- Readings
Logistics

 Meetings

• Mondays, 3PM – 5:20PM (M hour)
• Zoom

 Grading

✓ Paper reviews ➔ 10%
✓ Paper presentations ➔ 20%
✓ Discussion part. ➔ 20%
✓ Project report ➔ 40%
✓ Project presentation ➔ 10%

 Study material

■ No required textbook ➔ Assigned readings

 Communication

• https://cs.brown.edu/courses/csci2951-u/
• course.csci.2951u.2021-2022.s01@lists.brown.edu

 Check the website!

• Announcements
• Lecture slides
• Readings
Staff

▶ Instructor

Vasileios (Vasilis) Kemerlis
  • vpk@cs.brown.edu
  • https://www.cs.brown.edu/~vpk
Office hours: Mon. 6PM – 7PM (Zoom)
Figure 1. Attack model demonstrating four exploit types and policies mitigating the attacks in different stages.

- Make a pointer go out of bounds
- Make a pointer become dangling
- Use pointer to write (or free)
- Use pointer to read

VI.
Memory Safety

VII.A.
Data Integrity

V.B.
Data Space Randomization

VIII.A.
Code Integrity

VIII.B.
Data-flow Integrity

V.A.
Address Space Randomization

- Modify a data pointer
- Modify code ...
- Modify a code pointer ...
- Modify a data variable ...

- Output data variable
- Interpret the output data
- ... to the attacker specified code
- ... to the address of shellcode / gadget
- ... to the attacker specified value

VII.A.
Data Integrity

V.B.
Data Space Randomization

- Code corruption attack
- Control-flow hijack attack
- Data-only attack
- Information leak

- Use pointer by indirect call/jump
- Use pointer by return instruction
- Use corrupted data variable
- Execute available gadgets / functions
- Execute injected shellcode