COMP 4580 – Computer Security

Course Description

Calendar entry

Computer security and information management. This course will examine state-of-the-art knowledge about the issues relevant to data and computer security. Prerequisites: COMP 3430 and COMP 3010.

General Course Description

The need for security permeates every aspect of Computer Science. Every piece of software must do its part to ensure the safety of the computer its running on and the privacy of the people using it. In this course we will dive into security-first design as it pertains to the Operating System; the development of applications across desktop, mobile, and web platforms; and computer networks. Along the way we'll learn advanced techniques to secure and test the systems we use every day.

A nefarious actor will find the weakest link to exploit and compromise a computer and all the information found on it. This course will start you on your way to eliminating the weakest links in the systems you build and maintain.

Detailed Prerequisites

Before entering this course, a student should be able to:

- Understand what an operating system provides and how best to exploit that functionality in the code they write.
- Implement software that makes use of core operating system functionality.
- Manipulate memory buffers through pointers and address arithmetic.
- Perform well-defined and structured tests on code.
- Use a debugger to inspect program state and step through code line-by-line.
- Build their own client-server and peer-to-peer applications.
- Use messaging protocols such as HTTP.

Course Goals

By the end of this course students will:

- Make use of standard cryptographic tools including symmetric encryption, key-exchange protocol, and secure random number generation.
- Exploit common Operating System vulnerabilities and show how to protect against those exploits.

- Learn how to securely authenticate a person/process and control resource access based on that authentication.
- Apply secure programming techniques, such as defensive programming and input sanitization, to the implementation of an application.
- Be introduced to common network-based attacks and how to secure network traffic.

Learning Outcomes

Security Fundamentals

Students should be able to:

1. Define the pillars of security: confidentiality, integrity, availability, and authenticity.
2. Explain the different threat models and how they apply to each of hardware, software, communications, and data.
3. Differentiate between different types of malware.
4. Analyze the design of system with respect to security.

Applied Cryptography

Students should be able to:

1. Use a symmetric cryptographic algorithm (e.g., AES) within a cryptosystem that makes use of a secure chaining mode (e.g. CBC).
2. Use a Message Authentication Code to verify the sender of a message.
3. Generate a public key and authenticate the key using a Certificate Authority.
4. Explain how digital envelopes combine symmetric and public-key cryptosystems to secure communications (e.g., web sessions).
5. Explain the difference between statistically random and cryptographically secure random numbers.
6. Use a cryptographically secure random number generator to create unique keys.

Operating System Security

Students should be able to:

1. Explain how an Operating System supports the sharing of resources while also limiting access based on a process' permissions.
2. Explain and demonstrate how Operating System internals such as memory management, file systems, and driver management can be exploited to bypass access controls.
3. Describe common OS techniques (e.g., ASLR, canaries, guards, sandboxing) and how they protect an Operating System and the software making use of the OS.
4. Explain how build and distribution systems can be exploited to insert vulnerabilities into software systems.
5. Describe how build and distribution systems can protect against exploitation.

Access Control and Authentication

Students should be able to:

1. Describe how user- and data-oriented access control prevent unauthorized access to a resource.
2. Explain how an access matrix, access control lists, and capabilities access control mechanisms provide access control.
3. Describe the differences between authentication based on knowledge, possession, or biometrics.
4. Explain how to securely manage, store, and authentic a user based on a user ID and password.
5. Define two-factor authentication and design a system that securely uses 2FA using two of knowledge, possession, and biometrics.
6. Design and verify a remote authentication system using techniques such as OAuth, OpenID, and/or SRP.

Application Security

Students should be able to:

1. Apply defensive programming techniques to avoid common security pitfalls.
2. Rigorously test (e.g., through integration testing, fuzzing) their own code – and code, libraries, frameworks from other developers – for security vulnerabilities.
3. Implement input sanitization for a variety of input interpretation situations; including, but not limited to, SQL and browser-based applications.
4. Implement secure web applications to protect against attacks including, but not limited to, cross-site scripting.
5. Apply the rule of least privileges to the design and implementation of an application.
6. Make use of Operating System facilities to securely use features such as environment variables, temporary files, and error logging.

Network Security

Students should be able to:

1. Identify the security threats of different network layers and design techniques to address them.
2. Describe a variety of Denial-of-Service attacks and how each prevents/impairs authorized use of a system.
3. Explain how to detect and then respond to a DoS attack.
4. Explain how to preempt and/or prevent a DoS attack.
5. Design a firewall-based system that includes a set of rules for securing a network.
6. Identify the security limitations of a firewall-based system.
7. Explain how Virtual Private Networks provide authenticated and confidential communications over the public Internet.

8. Discuss the security implications inherent in wireless networking.