Course recap

Secure Programming
Lecture 20

In the news

http://krebsonsecurity.com/2014/03/sources-credit-card-breach-at-california-dmv/

Announcement

- Revision lecture
  - Tuesday 29/4, noon-1pm
  - UG05, Learning Centre
- Bring your own questions

Announcement

- I’m not going to be here from next week (most of the time)
- However, if something is not clear, send me an email
  - We can set up time to discuss things by voice (skype, google chat)

Plan for today

- Quick recap of what we saw
  - Key points
- Your questions

Goals

- Explain the fundamental principles and mechanisms of software security
- Identify the main security defects and threats in current software systems
- Describe and evaluate techniques of secure coding
- Evaluate applications in relation to their security

Fundamentals

- Vulnerabilities
- Exploits
- Risk
- Ethics consideration when researching this area

Principles

- Saltzer and Schroeder principles
  - Economy of mechanisms
  - Fail-safe design
  - Complete mediation
  - Open design
  - Separation of privilege
  - Least privilege
  - Least common mechanism
  - Psychological acceptability
  - Circumvention work factor
  - Compromise recording
- Other guidelines
  - Orthogonal security
  - Be skeptical, be paranoid
  - Design security in

Finding vulnerabilities

- Reviews
  - Design
  - Operation
  - Application
- Techniques
  - Attack trees
  - Risk rating
- Code audits strategies
- Fuzzing

SQL injection

- Root cause
- Defense: prepared statement
- Techniques to escalate vulnerability

More injections

- Command injection
- XPATH injection
- Serialization vulnerabilities
- More defenses
  - Sanitization
  - Static analysis

Buffer overflows

- Basics of Linux process (stack, procedure calls)
- Smashing the stack
  - RET overwrite
- Shellcode
  - NOP sleds
  - Creating and testing shellcode calling syscalls
  - Techniques: encoders, GetPC

Exploit mitigation

- Stack protection
  - Non-control-data attacks
- Address space randomization
  - Heap spraying
- Non-executable stack
  - Return into libc

Heap overflow

- Malloc design and implementation
  - dlmalloc: chunks, bins
- Abusing free chunk management to overwrite arbitrary memory locations

Race conditions

- Races in general
- Significant special case: Time of check to time of use (TOCTOU)
- Beating the odds

Other vulnerabilities

- Algorithmic complexity attacks
  - Attacks against hash tables
  - Attacks against regular expressions
- Integer overflow
  - Number representation
  - Common errors

Malicious web

- How are vulnerabilities exploited today
  - Drive-by-download attacks
- What happens after a vulnerability has been exploit
  - Botnets
- Underground economy

That’s all folks

- I hope you learned new approaches, techniques, ideas
- I hope you had fun in the process

YOUR QUESTIONS