CS 370 Introduction to Security **Week 8: Problem Set 8**
Instructor Name: Rakesh Bobba

Introduction/Purpose

The purpose of this assignment is to help you gain a better understanding and insight into software vulnerabilities such as buffer overflows other data and code injection attacks covered in Week 8

Before beginning make sure you have watched the lecture videos on the following and completed the associated practice quizzes.

- Common Software Vulnerabilities
- Input Handling Vulnerabilities
- Buffer Overflow I
- Buffer Overflow II
- OS Interaction Vulnerabilities

Chapter 4.4 from Security Engineering: A Guide to Building Dependable Distributed Systems by Ross Anderson

Instructions/Questions

Please answer the questions below.

Software Vulnerabilities and Injection Attacks

Q1 [4 pts] What is an injection attack? Give 2 examples of injection attacks

An injection attack is an attack refers to a broad class of attack vectors. An attacker supplies untrusted input to a program, such input gets processed by an interpreter as part of a command or query and in turns, alters the execution of the program. Two examples are SQL Injections and Cross-site scripting.

Q 2 [4 pts] Describe what a Cross-Site Scripting attack is. What type of an attack is this?

Cross-site scripting or XSS is an attack that results from attacker injecting an arbitrary script into a legitimate website or web application. The script is then executed inside the victim's browser. It is a type of injection attack.

Q3 [4 pts] What is SQL Injection? How can it be prevented?

SQL injection is a type of attack that attacker use to injects SQL statements that can read or modify database data.

It can be prevented by establishing which applications are vulnerable and then utilize parameterized database queries with bound, typed parameters and careful use of parameterized stored procedures in the database.

Buffer Overflow

Q4 [5 pts] What is a buffer overflow? What are the 3 distinct parts of process memory that buffer overflows typically target?

Buffer overflow is an anomaly where a program, while wiring data to a buffer, overruns the buffer's boundary and overwrites adjacent memory locations. Three distinct types of locations are stack, heap or data section of a process

Q5 [3 pts] At a high-level, what are the three steps to exploiting a buffer overflow? What are the possible consequences of a buffer overflow?

The three steps are to change the flow of control of injected code, inject code via overflow and execute injected code. Possible consequences of a buffer overflow are corruption of data used in the program, unexpected transfer of control in the program, memory access violations and eventual program terminations.

Q6 [3 pts] What is stack smashing?

Stack Smashing is the overflow when targeted buffer is located on stack, as a local variable in stack frame of a function and overwrite return address/frame pointer to address of attack code in memory

Q7 [5 pts] What is StackGuard and how does is protect against stack smashing attacks? Can it prevent stack smashing - why or why not? Is StackGuard a compile time or run-time defense?

StackGuard is a small patch for the GNU gcc compiler which enhances the way gcc generates the code for setting up and tearing down functions. Protect against stack smashing attacks by detecting and defeats stack smashing attack by protecting the return address on the stack from being altered. This is done by placing a canary word next to the return address on the stack. Once the function is done, the new tear down code from gcc first checks to make sure that the canary word is unmodified and intact before jumping to the return address. If the integrity of canary word is compromised, the program will terminate. It is a compile time defense.

Q8 [2 pts] What is an advantages of runtime defenses against compile-time defenses?

Run time defenses have executable address space protections which can block execution of code on stack and heap and guard pages, which introduces additional pages between critical regions of memory and mark them as illegal.

Q9 [3 pts] What properties should the *canary value* in StackGuard have? Integrity, unpredictability and different for each system

Q10 [5 pts] Name five mechanisms/ways you can think of to protect against buffer overflow attacks.

Five mechanisms that protect against buffer overflow attacks

1. Guard Pages
2. Address Space Randomization
3. Executable Address Space Protection
4. Safe Coding
5. Programming Language Choice
6. Extensions / Safe Library
7. Stack Protection

https://oregonstate.instructure.com/conversations

Q11 [2 pts] What is one advantage of Return Address Defender (RAD) over StackGuard?

Return Address Defender copy return address, have a safe location and does not make changes to stack structure.

Q12 [3 pts] What is the difference between StackGaurd and Guard Pages?

StackGuard is based on a canary value that is put on the stack with each function call and checking the canary at the end of the function while guard pages provides a one-shot alarm for memory access/

OS Interaction Vulnerabilities

Q13 [4 pts] What is a TOCTOU error? What is one way to prevent them?

Time-of-Check to time-to-use is an error that is caused by race condition involving the checking of the state of a part of a system and the use of the results of that check. It can be prevented by using exception handling and to use file locking to prevent race conditions for single.

Q14 [4 pts] What is a race condition? Why do they happen?

A race condition occurs when two or more threads can access shared data and they try to change it at the same time. It happens when one thread have a “check-then”act” case and does something while another thread do something to value between checks.

Q15 [4 pts] How can environmental variables be used for code injection?

Environmental variables is a way for the program to get input and is thus read by compiled programs and scripted programs. While it is usually set up, it can be injected prior to program read or prior to compile and adversary can thus use it to corrupt program. For example, using PATH or IFS to cause the script to execute attacker programs with privilege granted to scripts

Submission Details

Submit a PDF file with the questions and your corresponding answers

The assignment is worth 55 points. It is due Wednesday of Week 9 at Midnight.