Chapter 07 – Programming for Security Professionals

Notes

Overview

This chapter introduces basic programming concepts. It starts by explaining the C programming language syntax and its basic functions. Then, students will learn to create Web pages using HTML. HTML is a formatting language rather than a programming language. Next, the chapter describes Perl and shows how to create basic Perl scripts. Finally, you will be introduced to object-oriented programming, an alternative way to create computer programs.

Chapter Objectives

After reading this chapter and completing the exercises, you will be able to:

- Explain basic programming concepts
- Write a simple C program
- Explain how Web pages are created with HTML
- Describe and create basic Perl programs
- Explain basic object-oriented programming concepts

Tips

Introduction to Computer Programming

1. Just as book editors must understand the rules and syntax of the English language, computer programmers must understand the rules of programming languages and deal with syntax errors.

2. Most colleges don't teach programming with security in mind. Because of this, many attacks on OSs and applications are possible because of poor programming practices.

Programming Fundamentals

1. Learn the fundamental concepts of programming: branching, looping, and testing.

Branching, Looping, and Testing (BLT)

1. Understand the following concepts:
    a. Function
    b. Branching
    c. Looping
    d. Testing
    e. Algorithm
    f. Bug
    g. Pseudocode

2. The concepts above are the most basic concepts that you need to know to write or understand computer programs.

Tip	Find more information about writing pseudocode at: http://www.csc.calpoly.edu/~jdalbey/SWE/pdl_std.html

Documentation

1. Without proper documentation, it is difficult for programmers to understand their own work, even if it was only written a few weeks before. Documentation also helps others to understand your work. Even though you might find it a time consuming and tedious process, it is imperative that you document your code.

2. Good documentation helps programmers find bugs in their code.

Tip	Java has a standard for documenting Java code called Javadoc. Read http://www.oracle.com/technetwork/java/javase/documentation/index-jsp-135444.html for an overview of Javadoc.

Learning the C Language

1. C was developed by Dennis Ritchie at Bell Laboratories in 1972 as a powerful and concise programming language. C++ is an enhancement of the C language.

2. Understand the role of a compiler when developing applications.

Anatomy of a C Program

1. Viewing the “Hello, world!” sample program, understand the main components of a C program. Learn how to write single-line and multiple-line comments, load libraries, and functions (with and without parameters).

2. See Table 7-2 for some special characters that can be used when calling the printf() function.

Declaring Variables

1. Use of variables in your programs to represent numeric or string values. Variables must be declared before they can be used. Once a variable has been declared, it can be used in any part of the program.

2. See Table 7-3 for the variable types supported in C.

3. See Table 7-4 for some of the conversion specifiers that can be used in C. Conversion specifiers tells the compiler how to convert the values in a function.

4. See Table 7-5 and Table 7-6 for the mathematical and logical operators available in C.

Branching, Looping, and Testing in C

1. Branching in C is as simple as calling a function. Looping and testing can be achieved using three types of loops: while loops, do loops, and for loops.

2. While loops test the condition first and repeat some actions if the condition is true. Do loops repeat the actions first and then they test the condition. Do loops stop when the condition is false. For loops repeat some actions based on the value of a counter.

3. Reviewing the examples of code listed in the text, understand the construction of each of the loops covered in this section. See Figures 7-1 and 7-2.

Understanding HTML Basics

1. HTML is still the foundation language for Web development. As a security professional you need to know how to read HTML files and recognize when something looks suspicious.

Tip	You can visit http://www.w3schools.com/html/ for a complete HTML tutorial.

Creating a Web Page with HTML

1. A Web page consists of several HTML tags denote by the < and > symbols. Every HTML tag has a matching closing tag. It is a common mistake to forget the closing tag.

2. See Table 7-7 for some of the HTML formatting tags.

3. See Figure 7-4 for a sample of HTML source code. You can write HTML Web pages using any text editor such as Notepad in Windows.

Understanding Perl

1. Practical Extraction and Report Language (Perl) is a powerful scripting language used to write scripts and programs for security professionals.

Background on Perl

1. See Table 7-8 for the evolution of Perl. Perl was originally developed by Larry Wall in 1987 and it is supported by almost any platform.

2. Although hackers use Perl to create automated exploits and malicious bots, security professionals also use Perl to perform repetitive tasks and conduct security monitoring.

Understanding the Basics of Perl

1. Display and use the help documentation Perl command with perl -h.

2. The perldoc command displays detailed information of a perl scripting command. Note that Perl's syntax is very similar to C syntax.

Understanding the BLT of Perl

1. Every programming language must have a way to branch, loop, and test. Perl is no exception. See Perl code examples to learn how Perl handles these BLT functions.

2. Branching in Perl is as simple as calling a function in any portion of your code.

3. Looping in Perl can be accomplished using one of two mechanisms: the for loop or the while loop.

4. See Table 7-10 for Perl operators. Consider the example of combining operators with Perl conditionals.

Understanding Object-Oriented Programming

1. This section covers the fundamental concepts of object-oriented programming.

Components of Object-Oriented Programming

1. Viewing sample code, understand the basic anatomy of an object-oriented program. Classes are the main structures that hold pieces of data and functions. When writing an object-oriented program, you create classes to represent the type and behavior of an object. Objects are instances of classes. Note that you must use the object's name to invoke any of its member functions.

Tip	Check out http://gd.tuwien.ac.at/languages/c/c++oop-pmueller/ for an introduction to object-oriented programming using C++.

An Overview of Ruby

1. Another object-oriented language many security testers use is Ruby, which is similar to Perl.
2. See Figure 7-16 to see how to modify exploit shell code in Ruby.

Additional Resources

1. How to Program in C: http://www.cs.cf.ac.uk/Dave/C/CE.html
2. Programming with security in mind: http://www.dwheeler.com/secure-programs/
3. Java SE security: http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136007.html
4. Application Programming Interface (API): http://www.computerworld.com/article/2593623/app-development/application-programming-interface.html
5. Perl Debugging Tutorial: http://www.thegeekstuff.com/2010/05/perl-debugger/

Key Terms

- **algorithm**
- **assembly language**
- **branching**
- **bug**
- **class**
- **compiler**
- **conversion specifier**
- **do loop**
- **for loop**
- **function**
- **looping**
- **pseudocode**
- **testing**
- **while loop**