Chapter 9

BASIC OPERATING SYSTEM AND TCP/IP CONCEPTS

INTRODUCTION

A vulnerability in TCP/IP (the transmission control protocol/Internet protocol), continues to be a serious Internet security problem. The goal of this chapter is to provide you with a much better understanding of the real-world risks of TCP/IP reset attacks. In other words, to better understand the reality of this threat, the aim of this chapter is to provide some background into the basic workings of operating systems and of TCP/IP concepts, and then to build upon this foundation to understand how resets attacks work [1].

TCP/IP OVERVIEW

The first part of this chapter aims at providing some basic information about TCP/IP. Much of the detail has been intentionally glossed over, focusing primarily on that which is relevant to understanding TCP/IP reset attacks. If you already have a good understanding of TCP/IP, you may want to skip directly to the part of the chapter that discusses how reset attacks work. Please realize that this chapter only focuses on the aspects of TCP/IP necessary to understand reset attacks. If you don't already have a good understanding of TCP/IP, don't expect to be an expert after reading this chapter [1].

TCP/IP is an abbreviation for the Transmission Control Protocol, defined in RFC 793 which was released in September of 1981. TCP/IP is a connection oriented protocol that can reliably get information from one host to another across a network/Internet. By reliable, TCP/IP guarantees all data will arrive uncorrupted at the remote host, automatically detecting dropped or corrupted packets and resending them as needed. Every TCP/IP packet includes a header, which is defined by the RFC as shown in Fig. 9-1 [1].

Programs utilize TCP/IP by passing it buffers of data. TCP/IP breaks this data into packages known as segments, and then uses IP to further package these segments into

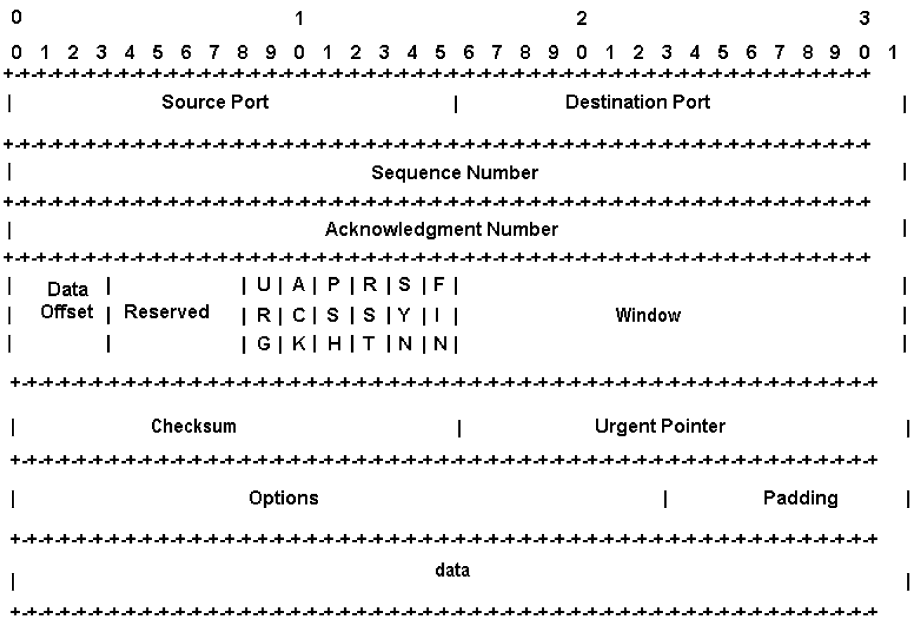

Figure 9-1. TCP/IP packet header.

datagrams. Finally, the datagrams are embedded into a network packet which can be routed across a network [1].

When the packet arrives at its destination, the IP stack on the remote host extracts the datagram from the packet, then the segment from the datagram. The segment is then passed up to the TCP/IP stack, where it can be validated. Ultimately the TCP/IP stack can reassemble all the segments into the complete buffer which is then passed to the application. TCP/IP provides two way communication, so this same process occurs in both directions [1].

Sequence Numbers

With data having been broken into segments which are then routed as separate packets over a network, it is quite possible for packets to arrive at their destination out of order. A field in the TCP/IP header provides for a 32-bit sequence number, a value that starts at an arbitrary integer then increments sequentially with each transmitted packet. Using these sequence numbers, the receiving TCP/IP stack is able to properly reorder the received segments [1].

Windows

TCP/IP also provides a mechanism for hosts to tell each other how much data they want to receive at a time. This is known as a window, defined by a 16-bit field in the TCP/IP header. Once defined, a host will generally only receive data that is within its specified window, dropping anything else. Being within the window means that the sequence number