Portscanner

Miro Bez Simon Targa

January 5, 2016

Contents

Introduction

The aim of this project was to implement a portscanner. The scanner should be written in C and support various scan methods (e. g. TCP connect scan, TCP SYN scan ...). The final result is a program that tries to simulate the behaviour of nmap, which is one of the most used programs for port scanning. This document describes how the program and the implemented scan methods work. The focus of the first section is on the TCP connect scan which is the most simple port scan technique. The second section is about how the TCP SYN scan works and how it was implemented within the scope of this project. Chapter three describes the scan methods Xmas, TCP NULL and Fin scan and their implementations. The fourth and final section is dedicated on how port scanning attempts can be detected and blocked by an IT administrator.

1 TCP connect scan

1.1 Theory

The TCP connect scan is probably the most easy method to scan for open ports. It simply takes advantage of the system call *connect* of the underlying operating system, to establish a connection with the target machine and port. Afterwards the returned value of the system call is used to determaine if the port to check is either closed or open at the target machine [1].

Extended description with details. Add advantages and disadvantages (table?)

1.2 Details of implementation

In order to use the system call connect the implemenation uses C sockets of the type SOCK_STREAM.

```
int mysocket;
mysocket=socket(AF_INET, SOCK_STREAM, 0);
```

Listing 1: Code to create C socket of type SOCK_STREAM

Continue to describe implementation

References

[1] Gordon Fyodor Lyon. *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Nmap Project, 2009. ISBN: 0979958717. URL: http://https://nmap.org/book/.