Sabbir Ahmed

CMSC 421: Project 2 Final Design Document

May 9, 2018

1 Introduction

This project implements a new version of the Linux kernel that adds functionality to support a simple intrusion detection system (IDS). This system will operate by logging the system calls made by a process in the kernel, while analysis and intrusion detection will be done in user space. This assignment is designed to teach a simple method of intrusion detection, as well as to reinforce the idea of how user space and kernel space interact through the use of system calls.

An intrusion detection system is a computer program that attempts to identify (and thwart) attacks that might be performed on the system by attackers. There are several time-tested approaches to the development of an IDS. The project will keep track of the system calls made by a monitored process and check for abnormalities in the sequences of system calls made. When an attacker breaks into a process, they will need to make system calls in order to attempt to access the resources of the system that are under attack. As the system calls that the attacker will perform will likely be different than those performed by a process that is not under attack, it follows that by monitoring both healthy and broken processes, it is possible to develop a scheme to identify those that might be under attack for further action to be taken.

2 Objective

The project will compare sequences of system calls made by a monitored process to known good sequences.

2.1 Kernel Space Requirements

The kernel-space program will instrument the system call dispatcher of the Linux kernel with code that logs each time a system call is made. Built-in system calls such as `ptrace` are not allowed to trace the usage of system calls

to generate the logs for the project.

2.2 User Space Requirements

The analysis of the logs will be handled by the user-space program, which may be implemented in any supported programming language. The user-space process should construct a bit array for each process under monitoring showing which system calls have been run in a window of the last k system calls. If a particular system call is made in the window, the bit for that system call will be set to 1. The bit arrays will then be measured for their hamming distance with the example of a healthy system call sequence for a process.