Building Recoverable Systems Using an Event Logger

Jessica Frame, Lawrence Phillips, Jonathan Torian, Julian Grizzard
{gtg473q, gtg688q, gtg749f}@mail.gatech.edu, grizzard@ece.gatech.edu

Introduction

In addition to preventing security breaches, it is equally important that the computer is able to operate efficiently. Thus, it is essential that computers be able to recover and continue running after being compromised by an attacker. In particular, preventing damage caused by compromises that use rootkits and other kernel editing approaches is observed. These rootkits leave a backdoor open to the computer so the attacker can visit and remain unseen.

Background

There are several possible ways to approach attempting to create a self-healing system. Being the most straight forward way to approach this problem, the following steps were chosen.

- Halt outside traffic to the computer.
- Repair the kernel.
- Restore internet.
- Send out a report of the attack.
- Continue normal functionality.

Method

A honeypot on a honeynet was chosen as the testing set-up for these self-healing machines because of several advantages:

- Safe to other machines.
- Created to record attacks.

Honeypots

In a honeynet, a transparent bridge sits between several systems, dubbed “honeypots.” This bridge has the ability to record traffic to and from the honeypots. To prevent systems from being used to attack other machines, the bridge blocks malicious outgoing traffic. Attacks are easily identifed because honeypots are passive.

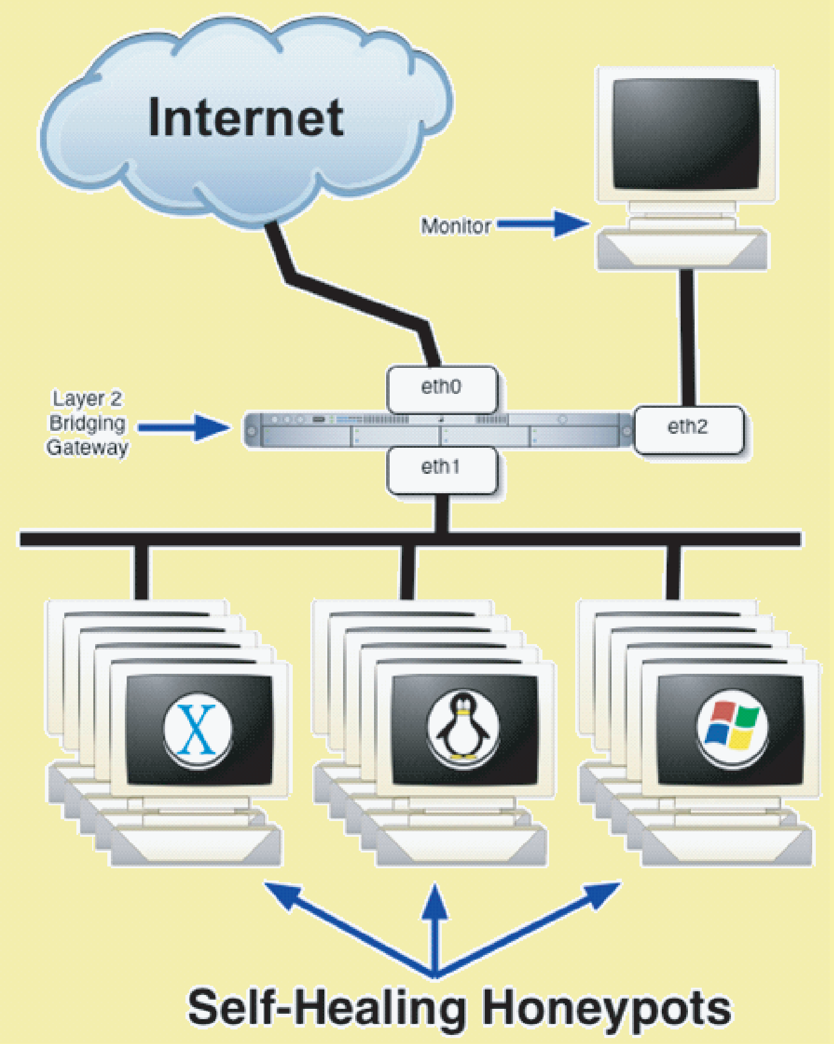

Figure 1: Logical Implementation of the Honeynet.

Rootkits

After gaining access to a machine, the attacker may leave a rootkit that leaves open a backdoor to the system and hide traces of the attacker’s illegal access for future visits to the system. The rootkit makes changes to program files such as ps, netstat, ifconfig, and other kernel utilities to hide attacker's presence.

While there are multiple types of rootkits, the presence of most rootkits is difficult to idenitfy. Thus, rootkits are a major computer security threat.

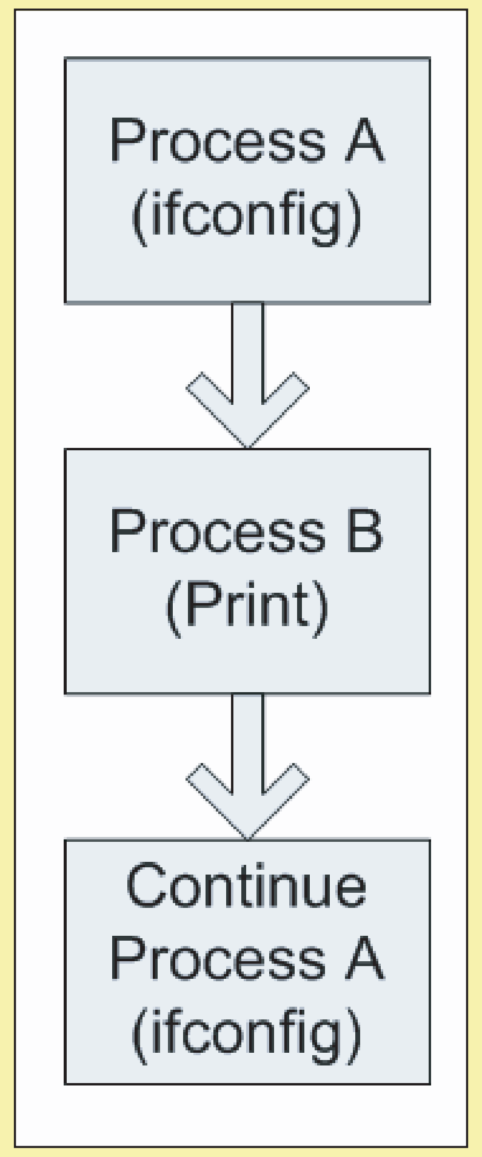

Figure 2: Normal flow of processes by the kernel.

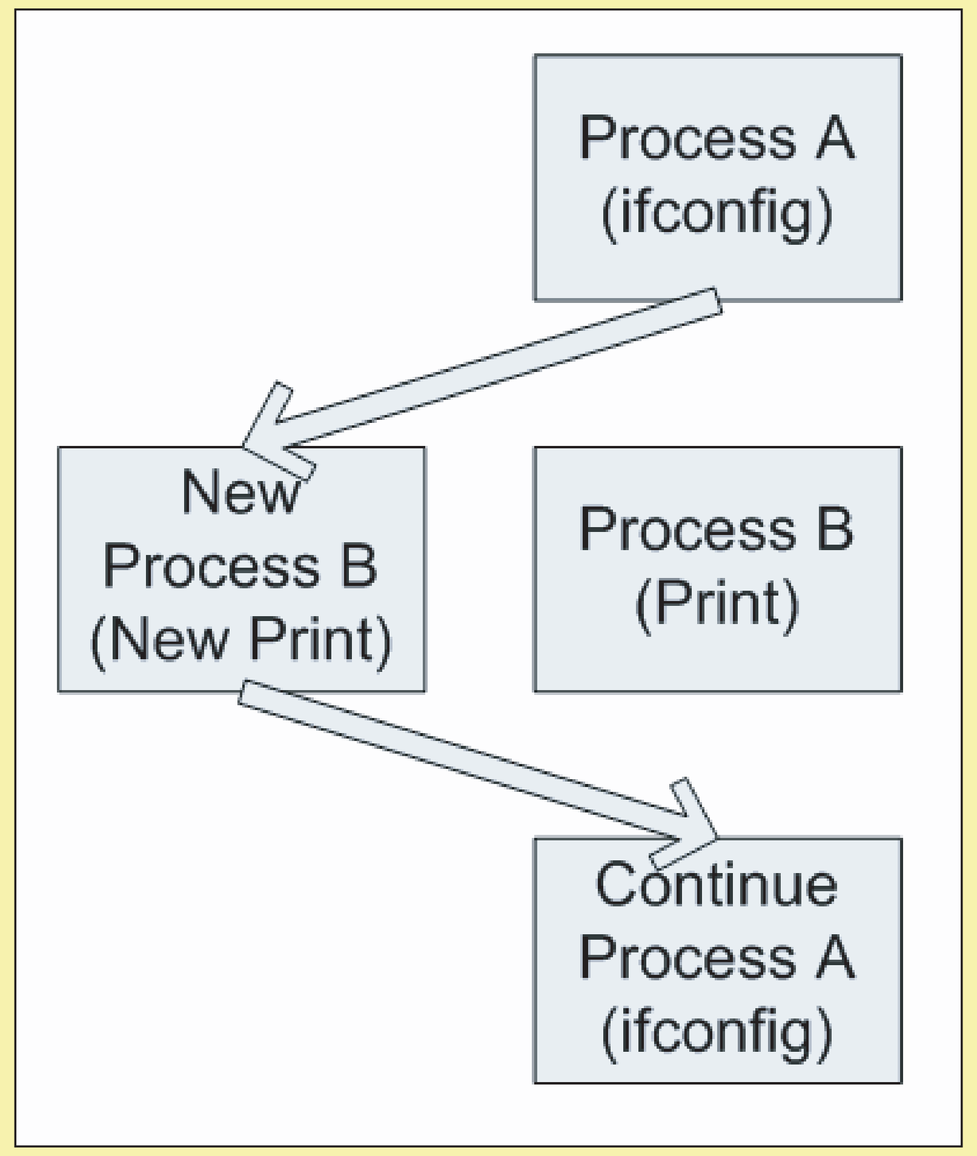

Figure 3: A possible flow of processes by a kernel with a rootkit.

L4 Architecture

The L4Linux kernel kernel runs on top of the L4 microkernel. Placing the event logger in the microkernel gives one an isolated view of everything being done by the L4Linux kernel.

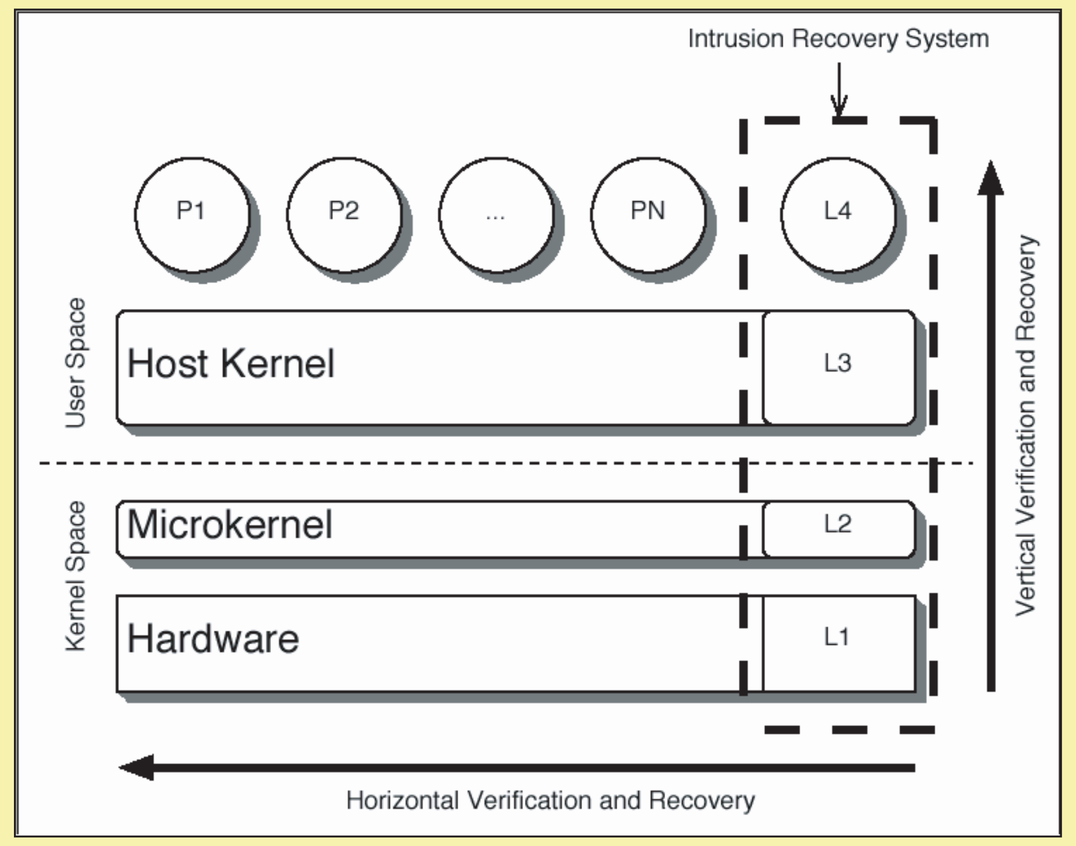
Intrusion Recovery System
P1
P2
...
PN
L4
User Space
Host Kernel
L3
Kernel Space
Microkernel
L2
Hardware
L1
Vertical Verification and Recovery
Horizontal Verification and Recovery

Figure 4: Diagram of layered computer flow model for intrusion detection

Future Work

- Test a complete self-healing system.
- Hide the event logger and self-healer programs from attackers.
- Secure the storage area for self-healer, event logger, and kernel repair information.
- Test the reporting system for attacks.