Honours Thesis (COMP4906)
Pre-Proposal & Permission Form

Year: 2019-2020

Student's Name: William Findlay

Student Number: 101015157

Student's Email: williamfindlay@cmail.carleton.ca

Supervisor: Dr. Anil Somayaji

Thesis Title: The Viability of eBPF for Intrusion Detection Implementations

---

In connection with your participation in COMP 4906, the School of Computer Science would like to publish (on the internet or otherwise), **your name**, your **supervisor's name**, the **title** of your Honours thesis as well as a brief text **abstract** describing your thesis. In order to do this, the privacy laws as contained in the **Freedom of Information and Protection of Privacy Act** or elsewhere requires that the school obtains your permission to do so. Please check one of the appropriate box below:

☑ I agree that such publication (as mentioned above) does not constitute a breach of privacy laws as contained in the **Freedom of Information and Protection of Privacy Act** or elsewhere, and waive all rights and remedies that would otherwise have accrued to me on account of such publication.

☐ I do not give permission to the School of Computer Science to make such information (as mentioned above) available to the general public (online or otherwise).

The school would like to use some images and video clips of interesting projects that students have worked on for the purposes of promoting the school (e.g., brochures, videos, webpages). If the school uses your images/videos, you will be identified as being the source of such media. Do you give the school permission to use your images/videos for such purposes ? **Yes** ☑ **No** ☐

Student Signature: _____

Supervisor Signature: _____

Submitted on: _____

Honours Thesis Pre-Proposal

The Viability of eBPF for
Intrusion Detection Implementations

William Findlay

August 21, 2019

1 Overview of Thesis

This section presents an overview of the thesis including motivation, objectives, research questions, and methodology.

1.1 Motivation

Modern computer systems are inherently complex; this complexity yields a plethora of opportunities for malicious actors to exploit such systems to their benefit. *Intrusion detection systems* seek to solve the problem of unwanted behavior through a variety of techniques, ranging from relatively simple heuristics, to neural networks.

Recent patches to the Linux Kernel have brought about *Extended Berkeley Packet Filtering* (*eBPF*), a technology which offers a myriad of prospects for use in intrusion detection systems. eBPF programs run in an isolated virtual machine, which means that they are able to hook into kernel-level tracing APIs while offering unprecedented stability – an eBPF program is guaranteed to never crash the kernel, and a built-in verifier prevents the loading of eBPF programs which could potentially damage the system or produce unwanted side effects. Using eBPF, it is possible to build a robust, portable, and future-proof intrusion detection system that leverages existing techniques and empowers them with a highly performant, modern, and safe kernel tracing API.

Process Homeostasis (*pH*), introduced in a 2002 dissertation by Dr. Anil Somayaji, is a perfect example of an intrusion detection system that can greatly benefit from the boons offered by eBPF. While pH offered a highly effective and compelling solution at the time, it suffered from a lack of official upstream support and, as it was implemented as a patch for Linux 2.2, soon became outdated. The advent of eBPF presents a unique opportunity to bring pH back and allow it to compete with modern solutions – many of which entail a much higher overhead, as well as a higher propensity to yield false positives in practice.

The proposed system, styled as *ebpH* (*Extended Berkeley Process Homeostasis*), builds on its predecessor, maintaining core functionality while offering the advantages of eBPF, including improvements to stability, flexibility, and widespread adoptability thanks to the prevalence of eBPF in the modern Linux world.

1.2 Main Objectives

While eBPF offers great promise to the area of intrusion detection and indeed computer security as a whole, much yet remains unexplored with respect to its feasibility in such applications. As such, the primary goal of this project will be to ascertain the degree of viability (or lack thereof) of eBPF implementations for production-grade IDS software.

In order to test the its viability for such applications, eBPF will be used to implement *ebpH*, a program that attempts to encompass the primary functionality of the original pH software by Dr. Somayaji. This means that it should be able to construct and maintain system-wide profiles for every binary, as well as detect and notify the end user of anomalous system calls as they occur throughout the system. Additional work will be done to make small improvements and/or changes to the original design where necessary.

For further details regarding the testing of ebpH (including criteria for performance, efficacy, and overhead), please refer to Subection 1.4 of this document.

1.3 Research Questions

This thesis is primarily concerned with the viability of eBPF-based intrusion detection systems. In particular, is such an implementation possible, how might it be done, and why is it worth doing? As such, the primary research questions which will be explored are as follows:

- What factors might contribute to eBPF being ideal for the implementation of an intrusion detection system?
  - ➢ Do other methods of implementation already provide these benefits?
- What caveats (if any) might exist which detract from eBPF's use for such implementations?
  - ➢ Is it possible to work around said caveats if necessary?
- How might an eBPF-based intrusion detection system perform (see Subection 1.4)?

1.4 Methodology

In order to ascertain whether eBPF is a viable method for implementing an intrusion detection system, ebpH will be implemented and tested for the following:

- Coverage: What types of attacks can the system detect and/or prevent?
- Overhead: What performance penalties are observed while running under various loads?
- Limitations: Is there anything another implementation can do that an eBPF implementation cannot?
  - ➢ Conversely, can an eBPF implementation circumvent any limitations imposed by other implementations?

ebpH will be tested in a controlled, virtualized environment, using conventional software testing methods. Additional testing will be done in a production or quasi-production environment.

2 Bi-Weekly Schedule for First Term (Fall 2019)

Table 2.1 presents a bi-weekly schedule for the fall 2019 term. This term will be used primarily for background research, implementation of various eBPF programs, and developing a plan for testing in the winter term.

Table 2.1: A bi-weekly schedule for the fall 2019 term.

Bi-weekly Period	Task
Sep 04 - Sep 17, 2019	Background research on various intrusion detection methods.
Sep 18 - Oct 01, 2019	Background research on bcc (BPF compiler collection) and eBPF.
Oct 02 - Oct 15, 2019	Implementation of ebpH.
Oct 16 - Oct 29, 2019	Implementation of ebpH.
Oct 30 - Nov 12, 2019	Implementation of ebpH.
Nov 13 - Nov 26, 2019	Draft thesis proposal report.
Nov 27 - Dec 06, 2019 [a]	Finalize thesis proposal report.

[a] This period is slightly shorter to reflect the end of the term.