# SOFTWARE REQUIREMENTS SPECIFICATION

# for

# Flareoff - Firewall as Linux Kernel Module

**Version 1.0 approved**

**Prepared by Tushar Choudhary**

**KIIT Deemed to be University**

**April 26, 2020**

# Contents

# Revision History

| Name | Date | Reason For Changes | Version |
|---|---|---|---|
| Flareoff | 25 April 2020 | Initial | 1.0 |

# 1 Introduction

## 1.1 Purpose

The purpose of this document is to present a detailed description of the open-source software Flareoff. It will explain the purpose and features of the software, the interfaces of the software, what the software will do and the constraints under which it must operate. This document is intended for users of the software and also potential developers.

## 1.2 Document Conventions

This document respects the IEEE standards.

## 1.3 Intended Audience and Reading Suggestions

- Typical Users, such as students, who want to use Flareoff for analyzing network packets, reduce or eliminate the occurrence of unwanted network communications while allowing all legitimate communication to flow freely.
- Advanced/Professional Users, such as engineers or researchers, who want to use Flareoff to filter incoming and outgoing network traffic based on their own preference.
- Programmers who are interested in working on the project by further developing it or to fix existing bugs.

## 1.4 Project Scope

Flareoff is a stateful software firewall implemented as a Loadable Kernel Module (LKM). Loadable Kernel Modules are kernel extensions that can be loaded into operating system kernel dynamically. Flareoff provides network security by filtering incoming and outgoing network traffic based on a set of user-defined rules.

## 1.5 References

Flareoff is an open-source software and is licensed under the GNU General Public License v3.0. Anyone interested is free to contribute to the project.
Flareoff on GitHub: `https://github.com/swingcake/flareoff`
GNU General Public License v3.0: `http://www.gnu.org/licenses/gpl.html`

# 2 Overall Description

## 2.1 Product Perspective

Flareoff was developed for everyone who is interested in network security and wants either to just experiment with it in order to understand it better or wants to use it as a means of filtering incoming and outgoing network traffic.
It is available as a Loadable Kernel Module (LKM) which are simply extensions to the basic kernel in your operating system that may be loaded/unloaded on the fly, without a need to recompile the kernel or reboot the system.

## 2.2 Product Functions

- Works as an efficient IP black/white list firewall.
- Allows programs to operate on specified ports.
- Blocks incoming traffic according to some predefined rules.

## 2.3 User Classes and Characteristics

- Typical Users, such as students, who want to use Flareoff for analyzing network packets, reduce or eliminate the occurrence of unwanted network communications while allowing all legitimate communication to flow freely.
- Advanced/Professional Users, such as engineers or researchers, who want to use Flareoff to filter incoming and outgoing network traffic based on their own preference.
- Programmers who are interested in working on the project by further developing it or fix existing bugs.

## 2.4 Operating Environment

While most if not all modern operating system include loadable kernel modules, this project has devoted almost all of its attention to Linux.
Module's code has to be recompiled for each version of the kernel that it is linked to. Modules are strongly tied to the data structures and function prototypes defined in a particular kernel version; the interface seen by a module can change significantly from

one kernel version to the next.
The kernel module is developed on and tested for the following Linux distribution:
Linux ubuntu 3.13.0-68-generic

## 2.5 Design and Implementation Constraints

Since Flareoff is essentially a LKM, it suffers from the same drawbacks as any other LKM does. The theory of how loadable kernel modules actually do their work is quite simple. The kernel keeps an array of pointers to the locations of system calls, so all a loadable kernel module has to do is replace pointers in that list with pointers to its own definitions of the same functions.
As an example, an LKM that wanted to prevent any other LKMs from being loaded would replace the system call to load a module with its own code that would essentially ignore the request, and possibly return an error. A more sophisticated version of the same LKM might make note of the name of the module, return success, and add the name of that module to any listing of all loaded modules, so all would seem as though the module had been loaded correctly.

## 2.6 User Documentation

Refer to Flareoff GitHub repository for setup instructions.
https://github.com/swingcake/flareoff
Feel free to open an issue for additional help.

## 2.7 Assumptions and Dependencies

Flareoff is developed in C. Kernel modules need to be compiled a bit differently from regular userspace apps, usually with the help of Makefile.
Fortunately, the make utility along with any other dependencies (build-essential and kernel headers) are already shipped with bare-bones Ubuntu.

# 3 External Interface Requirements

## 3.1 User Interfaces

Flareoff offers command-line interface for easy interaction through the built-in terminal in your Linux machine. It is also memory efficient and fast.

## 3.2 Hardware Interfaces

Minimum hardware requirements/specifications pose no concern as any modern day computer can easily power this lightweight software.

## 3.3 Software Interfaces

Flareoff can be up and running on almost any machine which is powered by the Linux kernel. However, it is tested only on Ubuntu so far.

## 3.4 Communications Interfaces

Flareoff requires internet connectivity to filter incoming and outgoing network traffic and reduce/eliminate the occurrence of unwanted network communications while allowing all legitimate communication to flow freely.

# 4 System Features

This section demonstrates Flareoff's most prominent features and explains how they can be used and the results they will give back to the user.

## 4.1 Port selection

Extension to the Linux firewall which makes it possible to specify which programs are allowed use which outgoing port.

### 4.1.1 Description and Priority

A firewall rule consists of a port number and a filename (the full path) of a program separated by a space, meaning that the corresponding program is allowed to make outgoing connections on this TCP-port. If there is no rule for a given port, any program should be allowed to make outgoing connections on this port. A connection is not allowed when rules for the port exist, but the program trying to establish the connection is not in the list of allowed programs. If a connection is not allowed, it should be immediately terminated.

### 4.1.2 Stimulus/Response Sequences

The kernel module processes the packets and maintains the firewall rules, and displays the firewall rules via *printk* in */var/log/kern.log.*
For every rule that is configured. is the port number in decimal representation and is the full path to the executable.
When the kernel module is unloaded, the firewall extensions should be deleted.

### 4.1.3 Functional Requirements

TBD (To Be Determined)

## 4.2 Network Filtering

Block incoming traffic according to some predefined rules.

### 4.2.1 Description and Priority

The *main* module inspects every packet and based on the rules provided in the problem statement, makes the decision of whether to accept the packet or drop it.
Following are the rules implemented:

- Block all unsolicited ICMP packets coming in from outside except the ones going to the web-server. However, the local hosts should be able to ping outside.
- Block all ssh attempts from outside.
- Block port 80 (HTTP) access from outside except for the web-server and test that an internal website on a local host is only accessible from inside.

### 4.2.2 Stimulus/Response Sequences

The incoming network traffic is blocked according to the rules provided.

## 4.3 IP black/white listing

Works as an efficient IP black/white list firewall.

# 5 Other Nonfunctional Requirements

## 5.1 Performance Requirements

Flareoff can be up and running on almost any machine which is powered by the Linux kernel. However, it is tested only on Ubuntu so far. Minimum hardware requirements/specifications pose no concern as any modern day computer can easily power this lightweight and efficient software.

## 5.2 Safety Requirements

This project *can be* harmful to your machine since it is essentially modifying the kernel, that is why it is always advised to set it up on a virtual machine. Continuous updates are being pushed by the developer to keep Flareoff as stable as possible. Users can raise an issue in the GitHub repository and pull requests (PRs) are most welcome.

## 5.3 Security Requirements

User requires *root* privileges to run Flareoff. It is required to load/unload the kernel module.

## 5.4 Software Quality Attributes

Flareoff provides the users with both basic and advanced features. Due to its well documented and easy-to-setup instructions, it can be used by both enthusiasts and typical users.

However, users must have a basic knowledge of Linux (kernel) before using it, and a desire to troubleshoot if things go south.

It is robust and portable and is readily available as an open-source software licensed under the GNU General Public License v3.0 on GitHub.

# 6 Appendix

## 6.1 Appendix A: Glossary

References: https://en.wikipedia.org/wiki/Main_Page

- **Firewall**: A firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules. A firewall typically establishes a barrier between a trusted internal network and untrusted external network, such as the Internet.
- **Kernel**: The kernel is a computer program at the core of a computer's operating system with complete control over everything in the system. It is an integral part of any operating system. It is the "portion of the operating system code that is always resident in memory". It facilitates interactions between hardware and software components.
- **Loadable Kernel Module (LKM)**: Kernel modules are pieces of code that can be loaded and unloaded into the kernel upon demand. They extend the functionality of the kernel without the need to reboot the system.
- **GNU General Public License**: The GNU General Public License (GNU GPL or GPL) is the most widely used free software license, which guarantees end users (individuals, organizations, companies) the freedoms to use, study, share (copy), and modify the software.

## 6.2 Appendix B: Analysis Models

Data flow diagram and use case diagram available separately.