Network Security Project
Implementation Details

Nicholas Capo - Nicholas.Capo@Gmail.com

March 2, 2012

Contents

1 Requirements Analysis

1.1 Language and Binary

1. The program shall be referred to herein as *Viper*
2. One version of the program shall be produced using the C language
3. One version of the program shall be produced using the Python language
4. Each version shall be compiled into two binaries (`viper` and `viper-test`) with the following usage:
   (a) `viper [ -h | --help ] [ -e | -d | --encrypt | --decrypt ] [ -t | --threads NUM ] [ -k | --key KEY ]`
   (b) `viper-test [ -h | --help ] [ -e | -d | --encrypt | --decrypt ] [ -k | --key KEY ] input_block`

1.2 Input/Output

1. `viper` shall expect input on `stdin`, and generate output on `stdout`
2. `viper-test` shall expect a single block of 32 hexadecimal values as the last argument on the command line
3. `viper` shall be the general case of `viper-test` and shall encrypt or decrypt until reaching end–of-input
4. All errors and help texts shall be written to `stderr`

1.3 Compatibility

1. Each version of `viper` shall be ciphertext compatible with the reference implementation of `Serpent`

1.4 Threading

1. Each version of `viper` shall implement a single threaded **TODO**
2. Each version of `viper` shall implement a multi-threaded **TODO**

2 Design

2.1 Overview

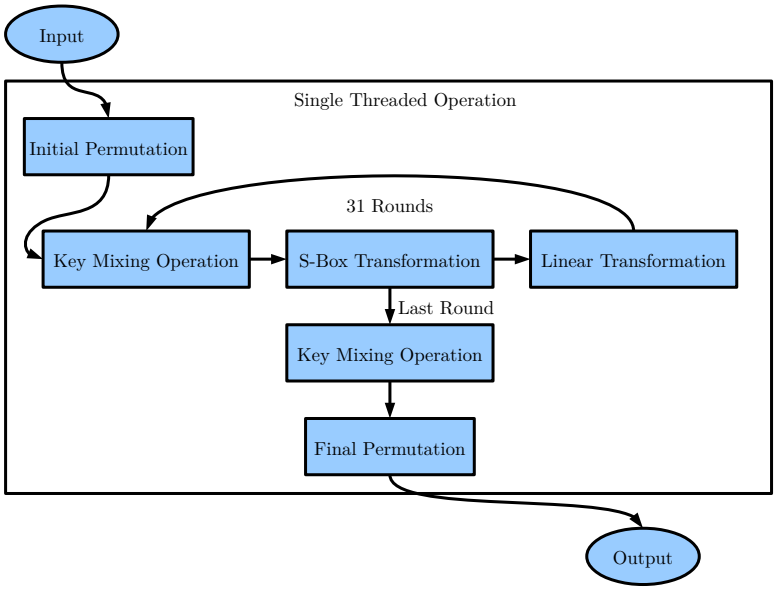

Figure 1: Cipher Dataflow Diagram

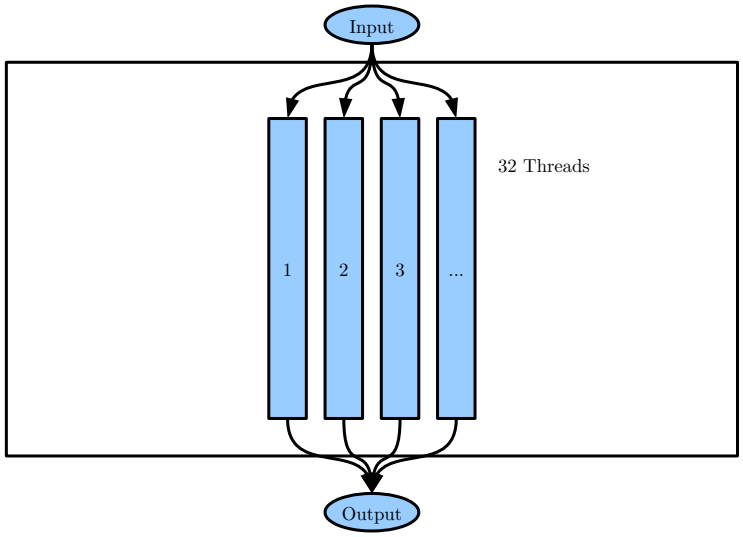

Figure 2: Threaded Dataflow Diagram

3 Code

4 Test Methodology and Results