Open source routing software
A comparative study of open source software routers

Bachelor Degree Project in Computer Science
Level ECTS
Spring term 2014

Fredrik Jakobsson

Supervisor: Jiangou Ding
Examiner: Jonas Mellin
Abstract

As the performance of PCs is increasing it is of great interest to use these cheap devices as routers, which traditionally consisted of more expensive and customized hardware for that purpose. The software was also traditionally proprietary and thereby costly, but as the open source community has grown there have been development of open source solutions that can perform the task of acting as a router. However as there are so many solutions out there, it can be hard for the potential users to choose which particular solution to use, without having to put in too much work into getting a fully functional router solution. This study achieved this purpose by benchmarking the most popular open source software routers, in terms of performance and scalability as well as providing a brief analysis of their basic security features. The routers that were studied was ClearOS, Untangle NG Firewall and IPFire, and after the study was complete IPFire was considered the superior with ClearOS as the second and Untangle as third and last.

**Keywords:** open source, software routers, ClearOS, Untangle NG Firewall, IPFire
Table of Content

1 Introduction ........................................................................................................................................... 1
2 Background ........................................................................................................................................... 2
  2.1 Routers ........................................................................................................................................... 2
  2.2 Open source software ..................................................................................................................... 2
  2.3 Open source software routers ........................................................................................................ 2
    2.3.1 ClearOS ...................................................................................................................................... 2
    2.3.2 IPFire ......................................................................................................................................... 4
    2.3.3 Untangle NG Firewall ................................................................................................................ 5
  2.4 Previous work .................................................................................................................................... 6
3 Problem definition ................................................................................................................................... 7
  3.1 Purpose ............................................................................................................................................ 7
  3.2 Motivation ......................................................................................................................................... 7
  3.3 Goals ............................................................................................................................................... 7
4 Methodology ........................................................................................................................................... 8
  4.1 Experiment ....................................................................................................................................... 8
  4.2 Theoretical study ............................................................................................................................ 8
  4.3 Validity ............................................................................................................................................ 8
  4.4 Alternative methods ....................................................................................................................... 9
  4.5 Motivation ....................................................................................................................................... 9
5 Implementation ....................................................................................................................................... 10
  5.1 Lab environment ............................................................................................................................ 10
  5.2 ClearOS .......................................................................................................................................... 10
  5.3 IPFire .............................................................................................................................................. 10
  5.4 Untangle NG Firewall .................................................................................................................... 10
  5.5 Tests .............................................................................................................................................. 10
    5.5.1 Throughput ............................................................................................................................. 11
    5.5.2 Latency ..................................................................................................................................... 11
    5.5.3 Frame loss rate ......................................................................................................................... 11
  5.6 Theoretical study ........................................................................................................................... 12
6 Results .................................................................................................................................................... 13
  6.1 Throughput ..................................................................................................................................... 13
  6.2 Latency .......................................................................................................................................... 15
  6.3 Frame loss rate ............................................................................................................................. 17
  6.4 Security features ............................................................................................................................ 19
    6.4.1 Firewall ..................................................................................................................................... 19
    6.4.2 VPN ......................................................................................................................................... 19
    6.4.3 Intrusion prevention ................................................................................................................. 20
  6.5 Comparison to related work .......................................................................................................... 20
7 Analysis .................................................................................................................................................. 21
8 Conclusion .............................................................................................................................................. 23
9 Discussion .............................................................................................................................................. 24
10 Future work........................................................................................................................................... 25

Appendix A – Network set up
Appendix B – Throughput – Results
Appendix C – Latency – Results
Appendix D – Frame loss rate – Results
1 Introduction

As the performance of PCs are improving and the open source community is growing, the interest in using an open source routing solution instead of a more traditional proprietary solution has increased. There are a multitude of available solutions on the market, but the most popular ones IPFire, ClearOS, and Untangle NG Firewall, are the ones being compared in this study.

This study aims to compare these most popular open source software routers’ most default configurations and provide a result that can be used by companies and individuals when making decisions regarding what solution they choose.

Experiments were done through benchmarking using the tools iperf and Ping and the results of the different softwares were compared to each other. A theoretical study was also performed to determine if the solutions provides with basic security features that are expected from a router.
2 Background

2.1 Routers

Routers initially consisted of equipment intended for the purpose of routing and that purpose alone and have so far been able to cope with the increase of demanded bandwidth. However, as they are lacking a de-facto standard regarding the hardware, which the personal computers (PC) have, the routers have been custom made by their manufacturers and the different vendors are thereby incompatible. A solution to this, is the implementation of software routers on PCs, which has been enabled by the open-source software community and the increasing performance of the PCs. A benefit of software routers, is a wide array of different hardware from different vendors to choose from, in difference from the earlier proprietary. This solution was initially discarded due to limited performance, instability of the software, lack of support, scalability issues, and lack of functions. Thanks to Moore’s Law, and the uprising of the open-source software community, these issues has been dealt with (Bianco et al., 2005).

2.2 Open source software

A software that is open source is not only free for everyone to use, but also available for everyone to use as a part of their own solution for free. That means that a developer can borrow source code from one open source software and use it as a part of its own software. All open source software must also be easily obtained and modified by all. It may also not be too bound to be used on a specific platform, it has to be able to be used by as many as possible (Open Source Initiative, 2014).

By implementing a routing structure using open-source software routers the costs of licenses, and specialized hardware that comes with the usage of proprietary solutions, can be avoided. Developers can also create their own required solution and support is available from the open-source software community (Wheeler, 2007).

2.3 Open source software routers

To act as routers there are a wide array of different open source software routers available and currently the most popular ones are ClearOS, FreeBSD and Untangle NG Firewall (Distrowatch, 2014b).

2.3.1 ClearOS

ClearOS Enterprise 6.5 is designed to be used as either a server, network, or gateway platform in a small business or distributed enterprise environment. It is based on a ClearOS core, which is a rebuild of Red Hat Enterprise Linux (Distrowatch, 2013). After the initial installation, the router can be configured from a remote web browser through the Webconfig (ClearCenter, 2014u) and besides the most basic services that are mentioned in this study, a wide array of additional services are available from the Marketplace (ClearCenter, 2014k).

The requirements of the hardware can be roughly estimated based upon the content of table 2.1.
<table>
<thead>
<tr>
<th>RAM and CPU</th>
<th>5 users</th>
<th>5-25 users</th>
<th>25-50 users</th>
<th>50-250 users</th>
<th>250+ users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor/CPU</td>
<td>Low-Power</td>
<td>Basic</td>
<td>Dual-Core</td>
<td>Quad-Core</td>
<td>Multi-Core + Multi-Processor</td>
</tr>
<tr>
<td>Memory/RAM</td>
<td>1-2 GB</td>
<td>2-4 GB</td>
<td>4-8 GB</td>
<td>8-16 GB</td>
<td>16-32 GB</td>
</tr>
</tbody>
</table>

*Table 2.1; Hardware Requirements (ClearCenter, 2014t)*

It can be run in three modes, where the last of the options are of interest in this study:

1. Standalone mode – No firewall
2. Standalone mode
3. Gateway

The two first options are used when ClearOS is used as a server and the difference of the two is the usage of a firewall, where the first option is suitable for a server in a trusted network, the second when used for public access, such as a web server and gateway is used when it is used as a router. There are different roles that can be set for the network interface when in gateway mode:

1. External
2. *Local Area Network* (LAN)
3. Hot LAN
4. *Demilitarized Zone* (DMZ)

Role 1 is for interfaces that connects the router to the Internet and there is the possibility to have multiple external interfaces for load balancing and automatic fail-over. The second role is for connecting trusted devices, such as desktops, using a address range of private IP addresses and role 3 is similar to the second, with the difference that is used for untrusted devices, such as public web servers. Devices in the Hot LAN are able to access the Internet, but not other devices on a LAN. If the case is that several public IP addresses are available and are to be used for public devices, such as a public web server, the DMZ-role is chosen (ClearCenter, 2014i). The last role is similar to the Hot LAN in the way that it is isolated from devices on LAN (ClearCenter, 2014d). Although there is not explicitly stated that many to many *Network Address Translation* (NAT) is supported, one might assume that it is activated and configured automatically as it is mentioned that devices in LAN and Hot LAN are granted Internet access, although they use private IP addresses.

ClearOS offers a basic *Dynamic Host Configuration Protocol* (DHCP) server, where pools are created and assigned with IP addresses and all other required parameters (ClearCenter, 2014c) and it is also possible to implement a *Network Time Protocol* (NTP) server (ClearCenter, 2014n). Other services that can be implemented are *Remote Authentication Dial-in User Services* (RADIUS) server (ClearCenter, 2014q) and *Secure Shell* (SSH) server (ClearCenter, 2014r).
In ClearOS it is possible to implement basic firewall functions, such as blocking all or certain traffic, based on port or IP address, from entering from the Internet or leaving for the Internet (ClearCenter, 2014f; ClearCenter, 2014h). If desired, it is possible to implement more advanced firewall rules with the use of iptables (ClearCenter, 2014b). It is also possible to use port forwarding to make a service with a private IP address publicly accessible from the Internet (ClearCenter, 2014o).

In order to provide Quality of Service (QoS), a bandwidth manager is available that limits and prioritizes traffic based on IP addresses and ports (ClearCenter, 2014a).

There are five different implementations available for VPN (Virtual Private Networking):

1. Dynamic VPN
2. ibVPN
3. OpenVPN
4. PPTP Server
5. Static IPsec VPN

Dynamic VPN is a solution that only requires that you create a VPN-connection between devices or users and then another party (Clear SDN) manages it for you, but in order to use this service you have to pay 100$/year (ClearCenter, 2014e). ibVPN is a service that allows users outside of the United States to access content limited to the United States, but it is not free of cost (ClearCenter, 2014g). Options that are free are OpenVPN (ClearCenter, 2014m), PPTP (ClearCenter, 2014p), and Static IPsec VPN (ClearCenter, 2014s), which all offer the use of encrypted tunnels for site-to-site VPN-connections and VPN-connections for remote users.

It is possible to have multiple WAN-connections (Wide Area Network) through the use of Multi-WAN, which enables to use load balancing and automatic fail-over (ClearCenter, 2014l). This can also be used to set up an environment of several routers using ClearOS to simulate the core of the Internet and test its routing capabilities. However, ClearOS has no support for any dynamic routing, so the routing tables has to be configured statically (ClearCenter, 2014j).

2.3.2 IPFire

“IPFire is a Linux distribution that focuses on easy setup, good handling, and high level of security. (Distrowatch, 2014a)”. The network interfaces can be set it four different modes, which are color coded: red for WAN, green for LAN, orange for DMZ, and blue for WLAN (Wireless LAN) (IPFire, 2014a).

It provides basic services such as SSH access for remote configuration (IPFire, 2014b), a DHCP server for all internally connected clients (IPFire, 2014c) as well as a NTP server (IPFire, 2014d).

The firewall of IPFire is a easy to manage Stateful Inspection Firewall that uses iptables in combination with a P2P filter (IPFire, 2014e).

For VPN connections, there are two available options: IPsec (IPFire, 2014f), and OpenVPN (IPFire, 2014g).
2.3.3 Untangle NG Firewall

Untangle is a Debian-based network gateway that uses applications to provide services such as spam blocking and intrusion prevention (Distrowatch, 2014c).

The following applications are available (Untangle, 2013j):

- **Web Filter;** monitors HTTP traffic to monitor user behavior and blocks inappropriate content (Untangle, 2014l).
- **Virus Blocker;** scans HTTP, FTP and SMTP traffic for malware and blocks it (Untangle, 2014j).
- **Spam Blocker;** filters email containing spam (Untangle, 2014i).
- **Phish Blocker;** protects user from phishing in email (Untangle, 2013e).
- **Web Cache;** caches HTTP content to save bandwidth when accessing previously accessed content (Untangle, 2013l).
- **Bandwidth Control;** monitors and controls bandwidth usage in the network (Untangle, 2014a).
- **Application Control;** controls traffic of individual applications such as file sharing and instant messaging (Untangle, 2013b).
- **HTTPS Inspector;** allows other applications in Untangle that process HTTP traffic to also process HTTPS traffic (Untangle, 2014d).
- **Captive Portal;** enables the requirement for users to log in or accept network usage policies before accessing the Internet (Untangle, 2014b).
- **Firewall;** blocks and flags traffic based on set rules (Untangle, 2013f).
- **Intrusion Prevention;** detects malicious activity in the network and if such is detected, its session is ended (Untangle, 2013g).
- **Ad Blocker;** blocks advertisement on web pages (Untangle, 2013a).
- **Reports;** used to provide the users with information regarding the traffic of the network (Untangle, 2013i).
- **Policy Manager;** provides the possibility to configure the applications to only apply to certain user groups and also to have different configurations for the different user groups (Untangle, 2014h).
- **Directory Connector;** enables Untangle to be able to communicate with directory servers, such as Active Directory (Untangle, 2014c).
- **WAN Failover;** if multiple WAN-connections are available it enables Untangle to maintain Internet connectivity if one goes down (Untangle, 2013k).
- **WAN Balancer;** if multiple WAN-connections are available it enables Untangle to distribute the traffic between the connections to maximize bandwidth usage (Untangle, 2014k).
• IPsec VPN; enables Untangle to use IPsec to create VPN-connections (Untangle, 2014e).

• OpenVPN; enables Untangle to create VPN-connections (Untangle, 2014g).

• Configuration Backup; enables Untangle to quickly be restored in case of hardware failures or disasters. It contacts a data center which performs the backup of the router (Untangle, 2013d).

• Branding Manager; allows the administrators to customize all user-facing components with company logos, name, URL and email (Untangle, 2013c).

• Live Support; allows the users to contact a support team via phone and email (Untangle, 2013h).

Untangle can also act as bridge, where multiple modes are available: standard bridge mode, DMZ bridge, and additional port. The first one extends the LAN, the second one extends the LAN into a DMZ and adds another LAN, and the third one is an extension of the first option, where an additional port is used to extend the LAN. NAT is also available to be used in Untangle, as well as VLAN and Virtual Redundancy Router Protocol (VRRP), which enables multiple Untangles to run in parallel to provide automatic fail-over (Untangle, 2014f).

2.4 Previous work

There have previously been done a similar study by Fahlesson (2013) that compared the, by that time, three most popular software routers to each other, where the set up consisted of a single software router, which was used to route packets between two clients. The conclusion from that study was that the tested software routers, Pfsense, ClearOS, and Vyatta performed equally, in terms of average throughput. Although there was a slight difference in throughput when testing packets of the size 64 bytes, where ClearOS was performing superior against the two others. ClearOS did however lack the support for dynamic routing protocols.

An earlier study by Guillen, Sossa and Estupiñán (2012) shows that the performance of the virtual software router Vyatta is better than a hardware router, due to the possibility to modify its physical features; its components.

A study by Rojas-Cessa Salehin and Egoh (2011) also shows that a regular software router (non-virtual) is superior to a virtual one as it is a single instance running on that physical machine. It thereby does not require to allocate computing power to enable the virtualization.
3 Problem definition

3.1 Purpose

The purpose of this study is to achieve a result that may be used by individuals or companies considering implementing a routing solution with open source software and the main question this study aimed to answer is:

*How well does the most popular open source routing solutions in their default configuration, perform in terms of performance, scalability, and security features?*

Open source software routers that were compared are ClearOS, IPFire and Untangle NG Firewall.

3.2 Motivation

As the performance of PCs are improving, it is highly motivated for companies and individuals to implement a routing solution that is open source instead of a proprietary one. There are a multitude of different possible solutions on the market and it can be hard for new users to choose which one to use, with the smallest of efforts, in order to have a fully operational routing solution.

The ones who may find use of this study are companies and individuals that are interested in or are planning to implement a routing solution consisting of open source software.

3.3 Goals

In order to answer the earlier given question of this study the following goals were to be met:

1. Benchmarking of the chosen routers, in terms of performance and scalability, and analyzing of the results.
   a) Performance in terms of throughput.
   b) Performance in terms of latency.
   c) Performance in terms of frame loss rate.
2. Determine if the chosen routers offer basic security features.
   a) Firewalls.
   b) VPN capabilities.
   c) Intrusion prevention.

These goals are prioritized in a top-down order and are therefore handled in the same order to make sure that the most important ones are completed. All goals are however, necessary to be completed in order for the study to yield a result that may be interesting to its stakeholders and comparable to previous studies.
4 Methodology

As Berndtsson et al. (2008, p.54) mentions, it is important to choose a proper method for achieving the objectives of the study and after considering multiple options, an experiment was chosen for goal 1 and a theoretical study for goal 2.

4.1 Experiment

For the experimentation, two PCs running a Linux distribution was used as clients where they were connected to one Ethernet interface of the router. The entire lab environment was isolated from any outer sources of disturbance, such as the Internet and other users. Tests were done according to Bradner and McQuaid (1999), where factors such as throughput, latency and frame loss rate were measured and the network topology used (appendix A) is also recommended by them. As the aim of the study is to compare the solutions in their default configurations, the only factor taken into account regarding the scalability is the frame size as other factors, such as queuing policies requires more advanced configuration. By performing the tests with different frame sizes, it generates results that show how well the routers scale with increasing frame sizes and the different sizes that were used for all test were the ones suggested by Bradner and McQuaid: 64, 128, 256, 512, 1024, 1280, and 1518 bytes. The tests with the purpose of measuring throughput and frame loss rate used the tool iperf (Iperf, 2013) and for the latency, Ping was used. All tests consisted of ten trials and after each trial, the routers were rebooted. Results were then analyzed in terms of minimum, average and, maximum values to determine, which router that was the superior. As Bouckaert et al. (2011) mentions that comparability is an important factor to take into account, all actions taken in the experiment was documented in order to support repeatability.

4.2 Theoretical study

Berndtsson et al. (2008, p.74) says that it is important to separate the useful sources of information from the less useful ones and with that in mind, the system documentations of the chosen software routers were chosen to be analyzed in the theoretical study. This was decided, as the system documentation is the source of information that is the most likely to contain the most recent updates of the software. Features that were looked into if they were available was: firewalls, VPN capabilities, and intrusion prevention. This information was then taken into account when making the final conclusions about how well the solutions compare to each other, as a solution lacking security features is not as good as one that has proper ones. The purpose of this stage was not to collect detailed information about the security features, but to find out if there were any available.

4.3 Validity

When performing experiments with computer software, there are multiple threats against the validity of the experiments and their results, that may occur and one that Wohlin et al. (2012) identifies is the issue of not being able to draw correct conclusions based on the results. It was handled by having each test consist of ten trials and by collecting the minimum, average and, maximum value. Wohlin et al. (2012) also mentions the issue of fishing after a certain result and it was handled by not expecting a certain result and having no favoritism towards a certain solution prior to the study and thereby the issue of fishing for a certain result was mitigated. The exact same tests were also done for each solution in order to avoid the possibility of this issue being brought up. Actions taken to generate results should be reliable and have the ability to be generalized and this was achieved in this study by having an
experimental design that is recommended by a RFC (Bradner and McQuaid, 1999). Besides the design of the experiment, the tools used for conducting the benchmarking was iperf and Ping, which are tools that are widely used for these kinds of tests. In order for a study to be valid, it is also important that the test objects are selected correctly to match the aim of the study and this was achieved by choosing the distributions that had the most hits on Distrowatch (2014b) and making sure that they were optimized to be used as routers. Another threat was outer sources of interference and that was handled by isolating the lab environment from any outer sources, such as the Internet and the LAN of the University of Skövde. The routers were also rebooted after each trial, in order to handle threats regarding the test objects having different uptime.

There are also threats, that Wohlin et al. (2012) mentions, that are not considered relevant for this study and some of these are the ones that are caused by test subjects. These are considered irrelevant as a quasi-experiment was performed and no subjects were thereby involved in the experiment. Threats that are caused by performing the experiment on different times are also ignored, as the experiment was performed in an isolated environment and no outer sources of interference, that varies depending on the time of the day was existing.

4.4 Alternative methods

A different approach that could have been taken would have been to perform a purely theoretical study, where other studies containing benchmarking results of the chosen solutions would have been analyzed. It should be noted that if the results of the study would have been based on other studies’ benchmarking, it is not sure that validity would be certain as the methods and actions taken to produce their results could differ from each other.

Another possible approach would have been to perform a case study, where the chosen solutions would have been monitored in a real life environment, but if that would have done it would have been hard, if not impossible to yield results that are valid. This would have been due to that the conditions, of the case being studied, would have varied from time to time and in turn it would have caused the results of the monitored solutions to be based on different variables.

4.5 Motivation

The chosen method was chosen as it provides a result that is free from validity issues as well as the experimental design itself. In the case of a solution that was lacking proper security capabilities is would also have been detected and would have affected the final verdict of the given solution.
5 Implementation

5.1 Lab environment

The platforms, which all routing solutions are tested on, are using the same hardware, which can be seen in table 5.1

<table>
<thead>
<tr>
<th>Component</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory/RAM</td>
<td>4 GB</td>
</tr>
<tr>
<td>Processor/CPU</td>
<td>Intel Core2 6400 @ 2.13GHz</td>
</tr>
<tr>
<td>Network/NIC</td>
<td>Supports network speeds up to 1 Gbits/s</td>
</tr>
</tbody>
</table>

Table 5.1; Hardware

The following sections explain how the particular software routers were installed and configured to provide the most basic configuration required to perform proper routing. This means that it supports the set up that can be seen in appendix A, where two Ethernet interfaces are used to interconnect two computers that perform the benchmarking.

5.2 ClearOS

Installed and initially configured locally on the computer through a graphical user interface (GUI) and afterwards accessed from a remote computer through a web-GUI, where the configuration was completed.

5.3 IPFire

Complete installation was done locally on the computer through a GUI, where basis settings were chosen and the network interfaces was configured to match the set up of appendix A.

5.4 Untangle NG Firewall

Installed and configured through a local GUI where a standard configuration was chosen with the most basic settings and no extra applications, i.e., only the network interfaces were configured so they would be able to route between the two subnets.

5.5 Tests

All tests were done with varying sizes as Bradner and McQuaid (1999) suggest; 64, 128, 256, 512, 1024, 1280, and 1518 bytes. Parameters that were measured were the throughput, latency, and frame loss rate. The benchmarking was done with the tools iperf and ping, where two test computers were required; one act as a server and one as a client for the process and each test had ten trials. Data from each test was collected and the minimum, average and maximum value from each test sequence was used for presentation of the results and used for the analysis of the study.
5.5.1 Throughput

The throughput of the routers was benchmarked by sending streams of packets from the client to the server with desired sizes and this was done with the tool iperf to prepare the computer used as a server the following command was issued:

iperf -s

This command tells the server to listen for TCP-traffic incoming on the default port (5001) and then the testing was done by using the following command on the client:

iperf -c 192.168.1.10 -l frame_size -t 600 -i 10 > result_file

The variables frame_size sets the size that the test run will use (e.g., 64 (byte)) and result_file that the results will be written to (e.g. test1). Time for each test to be run was set to 600 seconds and reports for each test was given each ten seconds so it would be possible to detect if the results varied throughout each test.

5.5.2 Latency

The latency of the routers were measured by sending packets from the client with the desired sizes with their maximum throughput and this was done with the use of the tool Ping by the following command on the client:

ping -c 120 -s frame_size-8 192.168.1.10 > result_file

It sends ten packets through the use of the flag -c 120 and the variable frame_size-8 (e.g., 48 (byte)) is the value of the desired size subtracted by eight as the ICMP header used by ping is of that size. The results are then written to the file given by result_file (e.g. test1) and ten tests were done for each size.

5.5.3 Frame loss rate

The frame loss rate of the routers was measured by sending a stream of packets with desired sizes and bandwidth from the client to the server and this was also done with the tool iperf were it was run in UDP-mode. Bandwidth used was the one earlier determined through the throughput testing.

Command that was issued to initiate the server to listen for UDP-traffic on the default port (5001) was:

iperf -s -u

Then the testing was done by entering the following command on the client:

iperf -c 192.168.1.10 -u -b throughput_for_given_frame_size -l frame_size -t 600 -i 10 > result_file

The flag -u tells iperf to send UDP-traffic instead of TCP-traffic, the variable throughput_for_given_frame_size (e.g., 35.3m (MB/s)) in combination with the flag -b sets the bandwidth that it will be sent with and result_file (e.g. test1) were the results will be written. Tests were run for 600 seconds each and reports were written each ten seconds.
5.6 Theoretical study

The system documentation of the chosen solutions was searched through in search of information regarding their security features, in terms of firewalls, VPN, and Intrusion prevention. All information about these features were then collected, analyzed and compiled into the study in a proper format.
6 Results

6.1 Throughput

The charts 6.1-6.3 displays the minimum, average, and maximum values for all the tested sizes of the tested software routers; ClearOS, Untangle NG Firewall, and IPFire. Chart 6.4 is then used to compare the average throughput of the software routers.

Chart 6.1; Throughput of ClearOS

As it can be seen in chart 6.1, the variation of the throughput in the different tests for ClearOS was very low; 9 Mbits/s in difference between minimum and maximum values at the two largest sizes. For all the remaining sizes, the variation was none.
In chart 6.2, which depicts the throughput of Untangle NG Firewall, it is visible that the variation of the minimum and maximum values were none until the size 1024 bytes, where it was a 15 Mbits/s difference, for 1280 bytes it was 12 Mbits/s and for 1518 bytes it was 15 Mbits/s.

**Chart 6.2; Throughput of Untangle NG Firewall**
In chart 6.2, which depicts the throughput of Untangle NG Firewall, it is visible that the variation of the minimum and maximum values were none until the size 1024 bytes, where it was a 15 Mbits/s difference, for 1280 bytes it was 12 Mbits/s and for 1518 bytes it was 15 Mbits/s.

The throughput of IPFire, which can be seen in chart 6.3, was steady; no variation in minimum and maximum values, until the size 1280 bytes was used. By then the variation was 11 Mbits/s and at the largest size, 1518 bytes, it was 9 Mbits/s.

**Chart 6.3; Throughput of IPFire**
The throughput of IPFire, which can be seen in chart 6.3, was steady; no variation in minimum and maximum values, until the size 1280 bytes was used. By then the variation was 11 Mbits/s and at the largest size, 1518 bytes, it was 9 Mbits/s.
Chart 6.4 shows that all solutions had the same average throughput for the sizes 64 (35.4 Mbits/s), 128 (70.7 Mbits/s), 256 (141 Mbits/s), and 512 bytes (283 Mbits/s). When the size was increased to 1024 bytes, Untangle slipped behind at 546 Mbits/s, while the two others had 566 Mbits/s. Untangle remained at that level for the remaining sizes, while ClearOS reached its limit at 1280 bytes with a throughput of approximately 530 Mbits and IPFire reached its as well, at about 674 Mbits/s.

6.2 Latency

The results of the tested solutions can be seen in chart 6.5-6.7 and in chart 6.8 a comparison of the routers’ average latency.

Chart 6.5; Latency of ClearOS
For Untangle (chart 6.6), the case was the same as for ClearOS; there was a slight variation of the latency’s minimum and maximum values and at most it was 0.049 ms, which was at the size 1024 bytes.

Chart 6.6; Latency of Untangle NG Firewall

As in for the others; there was a slight variation in minimum and maximum values for IPFire (chart 6.7), and at most it was 0.046 ms.

Chart 6.7; Latency of IPFire
IPFire had the lowest average latency for all sizes, which can be seen in chart 6.8. ClearOS came second and Untangle had the highest latency. This was the case for all the tested sizes.

### 6.3 Frame loss rate

The results of the individual routers can be seen in charts 6.9-6.11 and a comparison the routers average values is displayed in chart 6.12.

**Chart 6.8; Average latency**

IPFire had the lowest average latency for all sizes, which can be seen in chart 6.8. ClearOS came second and Untangle had the highest latency. This was the case for all the tested sizes.

**Chart 6.9; Frame loss rate of ClearOS**

For ClearOS (chart 6.9), the frame loss rate was close to zero for all sizes, except for 1518 bytes, where it was approximately 10%.
Untangle had a consistent frame loss rate for all sizes, which varied between 92 and 94%, which can be seen in chart 6.10.

**Chart 6.10; Frame loss rate of Untangle NG Firewall**
Untangle had a consistent frame loss rate for all sizes, which varied between 92 and 94%, which can be seen in chart 6.10.

IPFire (chart 6.11) had close to zero frame losses until the size 1280 bytes, where it was about 4.3% and even higher at 1518 bytes; varied between 22 and 23%.

**Chart 6.11; Frame loss rate of IPFire**
IPFire (chart 6.11) had close to zero frame losses until the size 1280 bytes, where it was about 4.3% and even higher at 1518 bytes; varied between 22 and 23%.
Untangle had a frame loss rate that averaged for about 93-94% for all sizes, which is visible in chart 6.12. ClearOS had close to zero until the largest size, where it had approximately 10% and IPFire had close to zero until 1280 bytes, where it had about 4% and at 1518 bytes it had about 22-23%.

### 6.4 Security features

#### 6.4.1 Firewall

ClearOS has two standard applications, *Incoming Firewall* and *Egress Firewall*, that is capable of blocking and allow traffic based on IP addresses and ports (ClearCenter, 2014f; ClearCenter, 2014h). In addition to this, it is also possible to use a more advanced firewall, *Custom Firewall*, which is retrievable from the Marketplace. This Custom Firewall uses iptables to create rules that are capable of more detailed blocking and allowing of network traffic (ClearCenter, 2014b).

Untangle NG Firewall also provides with a basic firewall that is capable of filtering network traffic based on IP addresses, ports, and protocols (Untangle, 2013f).

IPFire has a stateful inspection firewall that is a based of iptables and combined with a P2P filter, which means that is able to filter certain P2P protocols. *(IPFire, 2014e).*

In summary it can be said that all three solutions have the possibility to implement a solid firewall, as the software is available.

#### 6.4.2 VPN

For ClearOS there are several options when it comes to implementing VPN solution: Dynamic VPN, ibVPN, OpenVPN, PPTP, and Static IPsec VPN. The first option, Dynamic VPN, is a service that the vendors offer, but in return they want to get paid (ClearCenter, 2014e). ibVPN should not be considered a real VPN solution as it only allows users outside of the US to connect to services that exclusive to US residents, for a financial cost (ClearCenter,
However, the three last applications are free and all provide the capabilities expected from a VPN solution: encrypted site-to-site tunnels and remote-site connections (ClearCenter, 2014m; ClearCenter, 2014p; ClearCenter, 2014s).

Untangle has two options for setting a VPN solution: IPsec VPN (Untangle, 2014e) and OpenVPN (Untangle, 2014g). Both options are capable of site-to-site connections and remote-site connections.

IPFire has the same options as Untangle in terms of VPN, it supports IPsec and OpenVPN (IPFire, 2014f; IPFire, 2014g).

Based on that all solutions has the possibility to create a VPN solution it can be concluded that all of them ticks the box in terms of the VPN factor.

### 6.4.3 Intrusion prevention

ClearOS has intrusion prevention available through application. This application receives weekly updates of attack signatures. This is to increase its effectiveness and reduce the false positives (ClearCenter, 2014v).

Untangle has a similar application as ClearOS for intrusion prevention, which detects malicious activity in the network and terminates the session (Untangle, 2013g).

For implementing intrusion prevention, IPFire has the option to install Guardian, which is an addon for Snort that analyzes Snort's log files and blocks based on their content (IPFire, 2014h).

In conclusion, it can be said that all solutions have a possibility to implement intrusion prevention in some sort.

### 6.5 Comparison to related work

When comparing the results of this to studies that already exists, one can also make some assumptions, assumed that those studies are valid, such as that Fahlesson (2013) claims that ClearOS was superior Vyatta and Pfsense. If that is the case, then also IPFire is superior those. Guillen, Sossa and Estupiñán (2012) also claims that Vyatta is superior to any hardware router, as it has the possibility to easily improve its hardware components. With that claim in mind, one can also say that all tested routers of this study is superior any hardware router, as they all are open source software routers and it also applies to them, as their components are easily improved with a low cost.
7 Analysis

The results of the throughput benchmarking (chart 6.1-6.3 and appendix B) indicates that there were no dramatical variations in minimum and maximum values for either of the routers tested. As this is the case one can look at the average values of the results (chart 6.4) and start comparing the routers to each other. All routers tested had the same results without any major variations (a measly 0.1 Mbits/s at most) until the size 1024 bytes; 35.4 Mbits/s at 64 bytes, 70.7 Mbits/s at 128 bytes, 141 Mbits/s at 256 and 283 Mbits/s at 512 bytes. However at 1024 bytes, Untangle had about 546 Mbits/s while ClearOS and IPFire had 566 Mbits/s. As Untangle had approximately the same value for the next size, 1280 bytes, it indicated that it had reached its limit as the result for 1518 bytes showed the same thing. ClearOS and IPFire separated at 1280 bytes, where ClearOS had about 630 Mbits/s and IPFire had approximately 674 Mbits/s. These values were their limits as they had approximately the same results for the next and largest size, 1518 bytes. Based on these findings one can make the assumption that IPFire has the superior throughput as it had, from the point where they did not have equal results, a higher minimum value, in terms of throughput, than the other's maximum value. The same can be claimed for ClearOS in contrast to Untangle as the case was the same there; it had a higher minimum value then Untangle's maximum value, when they were not equal. Summarized it can be said that ClearOS had an approximate maximum throughput of about 630 Mbits/s, Untangle had 546 Mbits/s and IPFire had 674 Mbits/s.

Purely based on the average latency, which is displayed in chart 6.8, IPFire had the lowest values for all sizes and therefore the superior of the three, with ClearOS as second. But to get a more detailed view of the situation one has to look at the minimum and maximum values as well and how they compare, which can be seen in chart 6.5-6.7 and appendix C. For the smallest size, 64 bytes, IPFire had a maximum value of 0.205 ms, which was equal to ClearOS's minimum value and 0.01 ms lower than Untangle's minimum value (0.215 ms). That gives IPFire the best result for 64 bytes, but as for the second place, ClearOS gets it as it had lower minimum value as well as a lower maximum value than Untangle; 0.205 ms against 0.215 ms and 0.230 ms against 0.236 ms. The case was the same for 128 bytes; IPFire had a lower maximum value (0.199 ms) than the others' minimum values (0.2 ms for ClearOS and 0.214 ms for Untangle). Second place went once again to ClearOS as it had a lower minimum value than Untangle (0.2 ms against 0.214 ms), when both had equal maximum value (0.231 ms). For the size 256 bytes, IPFire had a maximum value of 0.207 ms, which was lower than the minimum values of ClearOS and Untangle; 0.216 ms and 0.223 ms, and that gave it the best results for that size as well. ClearOS once again came second as it had a lower minimum value than Untangle as well as a lower maximum; 0.236 ms against 0.245 ms. At 512 bytes IPFire had a maximum value of 0.212 ms, which was lower than ClearOS' and Untangle's minimum values that was 0.226 ms and 0.231 ms. The second place went to ClearOS as it had the lowest minimum value of the two as well as the lowest maximum; 0.240 ms against 0.248 ms. When the size was increased to 1024 bytes, IPFire had the lowest minimum and maximum values of the three; 0.227 ms and 0.255 ms compared to ClearOS' 0.235 ms and 0.278 ms as well as Untangle's 0.245 ms and 0.294 ms. As for the previous sizes, ClearOS came second again. At 1280 bytes, IPFire had a maximum value of 0.296 ms which was lower then ClearOS' and Untangle's minimum of 0.298 ms and 0.310 ms. That gave it the best results of that size as well and ClearOS had the second best as it the lowest minimum of the two remaining as well as the lowest maximum; 0.317 ms against 0.347 ms. For the last and largest size, 1518 bytes, IPFire once again had the lowest minimum and maximum values of the three; it had 0.328 ms and 0.374 ms, ClearOS had 0.350 ms and 0.391 ms and Untangle had 0.353 ms and 0.394 ms. Based on those numbers ClearOS came
When analyzing the results of each size in detail it can be concluded that IPFire also had the best results in terms of latency and ClearOS the second best.

When looking at the average frame loss rate in chart 6.12, it can be easily be said that Untangle had the worst results and ClearOS the best but in order to properly analyze the results one has to look at more data than just the average values. This data can be seen graphically in charts 6.9-6.11 and in appendix D. As Untangle had the highest values in general for all sizes; lowest minimum was 92% and highest maximum was 94%, it can be concluded that it had the worst results as the two others were not even close to those numbers. For the size 64 bytes both ClearOS and IPFire had a minimum value of 0% but ClearOS had a lower maximum value than IPFire; 0.0081% against 0.065%, which made that ClearOS had the best results for that size. As it increased to 128 bytes, both had a minimum of 0%, but ClearOS had a maximum of 0.012%, which was lower than IPFire's 0.069% and that once again made it have the best results. When size was set to 256 bytes ClearOS had a minimum value of 0.0043% compared to IPFire's 0.061% and maximum of 0.061% against 0.0680%. As ClearOS had lower values in both cases it had the superior results for that size as well. The case was the same for the sizes 512 and 024 bytes; both had minimum and maximum values close to zero but ClearOS had the lowest; for 512 bytes ClearOS had 0.0012% and 0.13% compared to IPFire's 0.0260% and 0.14% and for 1024 bytes ClearOS had 0.025% and 0.0750% and IPFire had 0.044% and 0.13%. However when the size 1280 bytes was used IPFire had a minimum and maximum of 4.3% while ClearOS still had numbers close to zero. But when 1518 bytes was used ClearOS had a minimum of 9.5% and a maximum of 10% against IPFire's 22% as minimum and 23% as maximum. Summarized from the analyzing of each size it can be concluded that ClearOS had the best results for each size and IPFire had the second best, while Untangle was clearly the inferior.

All solutions had a possibility to implement security features in terms of firewalls, VPN, and intrusion prevention. This means that the factor of security should not affect the result of the study and therefore only be dependent on the results of the benchmarking. However, one security feature that is included in the default configuration of Untangle is Shield, which blocks network traffic that it interprets as a DoS-attack. This was believed to be the cause of the extremely high frame loss rate of Untangle as there are several mentions on the official forums of Untangle regarding this issue. In order to provide results that would not have had that high frame loss rate, the service could have been disabled, but that would have caused the study to miss its aim, as it would have used a configuration that was not default. However, tests were done were Shield had been disabled, in order to verify that it actually was the cause and so was the case, as the frame loss rate had been decreased to levels comparable to the ones of ClearOS and IPFire.

Untangle was considered the worst of the tested routers, as it performed worst in all terms and did not scale as good as the others with increasing sizes; its throughput was limited at a lower value than the others at a smaller size than when the others reached their limits and its latency was consistently highest for all sizes. The router that was considered the best of the tested routers was IPFire, as it had the best results in throughput and latency, whereas ClearOS was superior in only frame loss rate. IPFire also scaled the best with increasing sizes, as it had the highest throughput at the two highest sizes, 1280, and 1518 bytes, when IPFire and ClearOS reached their limits. ClearOS however, had the best frame loss rate and scaled the best in that term, as it had a close to zero frame loss rate up and until the last size, where it had a 10% loss rate, whereas IPFire had a close to zero loss rate up and until the second largest size, where it had a 4.3% frame loss rate and an average 22.2% at the largest.

Regarding the scaling of latency IPFire was considered the best, as well as it had consistently the lowest latency for all sizes.
8 Conclusion

This study aimed to provide the differences in performance, scalability, and security features of the currently most popular open source software routers (ClearOS, Untangle NG Firewall and IPFire). Results was achieved by benchmarking in terms of throughput, latency, and frame loss rate with increasing sizes, as well as a theoretical study of the routers’ security features.

The results of the study showed that all routers being tested and analyzed were capable of offering options for basic security, such as firewalls, VPN, and intrusion prevention. In terms of the benchmarking, IPFire was considered the best performing with ClearOS as second and Untangle the worst, as it had the superior throughput and latency that also scaled the best. However, it did not have the best scaling frame loss rate, which ClearOS instead had. Also Untangle had a frame loss rate that was alarmingly high for all sizes and this is was due to a default security feature that interpreted the test as a DoS-attack and thereby blocked it.
9 Discussion

The theoretical study, which aimed to provide knowledge about the existence of basic security features of the routers, was very brief; essentially just found out about if they were capable of implementing solutions for firewalls, VPN, and intrusion prevention. A different approach that could have taken, could have been to perform a more detailed analysis of their security and performed benchmarking that applied to that as well, such as CPU usage when the firewall is actively working, i.e., sending traffic that is blocked by the firewall. The existing benchmarking could also have been extended to include more scalability testing, such as using different queuing policies.
10 Future work

After performing this study I have come up with some ideas for future work that could be performed within this subject. One of these, is a study that includes all active open source software routers and thereby the issue of including the correct test subjects would be eliminated. It would also provide with an easy way for all individuals interested in implementing an open software routing solution to get hold of the appropriate information, without having to go through multiple studies.

Another idea for a work could be to even include more general solutions such as FreeBSD, which would be optimized for routing, as they make out a large portion of open source software routers used in the world.
References


System documentation


Appendix A – Network set up
## Appendix B – Throughput – results

<table>
<thead>
<tr>
<th>Size</th>
<th>ClearOS</th>
<th>Untangle</th>
<th>IPFire</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>35.3</td>
<td>35.3</td>
<td>35.4</td>
</tr>
<tr>
<td>Min.</td>
<td>35.3</td>
<td>35.3</td>
<td>35.4</td>
</tr>
<tr>
<td>Avg.</td>
<td>35.4</td>
<td>35.4</td>
<td>35.4</td>
</tr>
<tr>
<td>Max.</td>
<td>35.4</td>
<td>35.4</td>
<td>35.4</td>
</tr>
<tr>
<td>128</td>
<td>70.6</td>
<td>70.7</td>
<td>70.7</td>
</tr>
<tr>
<td>Min.</td>
<td>70.6</td>
<td>70.7</td>
<td>70.7</td>
</tr>
<tr>
<td>Avg.</td>
<td>70.7</td>
<td>70.7</td>
<td>70.7</td>
</tr>
<tr>
<td>Max.</td>
<td>70.7</td>
<td>70.7</td>
<td>70.7</td>
</tr>
<tr>
<td>256</td>
<td>141</td>
<td>141</td>
<td>141</td>
</tr>
<tr>
<td>Min.</td>
<td>141</td>
<td>141</td>
<td>141</td>
</tr>
<tr>
<td>Avg.</td>
<td>141</td>
<td>141</td>
<td>141</td>
</tr>
<tr>
<td>Max.</td>
<td>141</td>
<td>141</td>
<td>141</td>
</tr>
<tr>
<td>512</td>
<td>283</td>
<td>283</td>
<td>283</td>
</tr>
<tr>
<td>Min.</td>
<td>283</td>
<td>283</td>
<td>283</td>
</tr>
<tr>
<td>Avg.</td>
<td>283</td>
<td>283</td>
<td>283</td>
</tr>
<tr>
<td>Max.</td>
<td>283</td>
<td>283</td>
<td>283</td>
</tr>
<tr>
<td>1024</td>
<td>566</td>
<td>546.2</td>
<td>566</td>
</tr>
<tr>
<td>Min.</td>
<td>566</td>
<td>546.2</td>
<td>566</td>
</tr>
<tr>
<td>Avg.</td>
<td>566</td>
<td>556</td>
<td>566</td>
</tr>
<tr>
<td>Max.</td>
<td>566</td>
<td>556</td>
<td>566</td>
</tr>
<tr>
<td>1280</td>
<td>629.9</td>
<td>548.7</td>
<td>674.3</td>
</tr>
<tr>
<td>Min.</td>
<td>627</td>
<td>542</td>
<td>670</td>
</tr>
<tr>
<td>Avg.</td>
<td>629.9</td>
<td>548.7</td>
<td>674.3</td>
</tr>
<tr>
<td>Max.</td>
<td>636</td>
<td>555</td>
<td>681</td>
</tr>
<tr>
<td>1518</td>
<td>627</td>
<td>545</td>
<td>670</td>
</tr>
<tr>
<td>Min.</td>
<td>627</td>
<td>545</td>
<td>670</td>
</tr>
<tr>
<td>Avg.</td>
<td>630.9</td>
<td>548.9</td>
<td>674.2</td>
</tr>
<tr>
<td>Max.</td>
<td>636</td>
<td>559</td>
<td>679</td>
</tr>
</tbody>
</table>
## Appendix C – Latency – Results

<table>
<thead>
<tr>
<th></th>
<th>ClearOS</th>
<th>Untangle</th>
<th>IPFire</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>64</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min.</td>
<td>0.205</td>
<td>0.215</td>
<td>0.176</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.219</td>
<td>0.224</td>
<td>0.187</td>
</tr>
<tr>
<td>Max.</td>
<td>0.230</td>
<td>0.236</td>
<td>0.205</td>
</tr>
<tr>
<td><strong>128</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min.</td>
<td>0.200</td>
<td>0.214</td>
<td>0.183</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.221</td>
<td>0.222</td>
<td>0.191</td>
</tr>
<tr>
<td>Max.</td>
<td>0.231</td>
<td>0.231</td>
<td>0.199</td>
</tr>
<tr>
<td><strong>256</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min.</td>
<td>0.216</td>
<td>0.223</td>
<td>0.184</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.228</td>
<td>0.235</td>
<td>0.195</td>
</tr>
<tr>
<td>Max.</td>
<td>0.236</td>
<td>0.245</td>
<td>0.207</td>
</tr>
<tr>
<td><strong>512</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min.</td>
<td>0.226</td>
<td>0.231</td>
<td>0.189</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.234</td>
<td>0.238</td>
<td>0.203</td>
</tr>
<tr>
<td>Max.</td>
<td>0.240</td>
<td>0.248</td>
<td>0.212</td>
</tr>
<tr>
<td><strong>1024</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min.</td>
<td>0.235</td>
<td>0.245</td>
<td>0.227</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.254</td>
<td>0.275</td>
<td>0.239</td>
</tr>
<tr>
<td>Max.</td>
<td>0.278</td>
<td>0.294</td>
<td>0.255</td>
</tr>
<tr>
<td><strong>1280</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min.</td>
<td>0.298</td>
<td>0.310</td>
<td>0.259</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.307</td>
<td>0.328</td>
<td>0.284</td>
</tr>
<tr>
<td>Max.</td>
<td>0.317</td>
<td>0.347</td>
<td>0.296</td>
</tr>
<tr>
<td><strong>1518</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min.</td>
<td>0.350</td>
<td>0.353</td>
<td>0.328</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.364</td>
<td>0.366</td>
<td>0.345</td>
</tr>
<tr>
<td>Max.</td>
<td>0.391</td>
<td>0.394</td>
<td>0.374</td>
</tr>
</tbody>
</table>
### Appendix D – Frame loss rate – results

<table>
<thead>
<tr>
<th>Frame Rate</th>
<th>ClearOS</th>
<th>Untangle</th>
<th>IPFire</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>64</strong></td>
<td><strong>Min.</strong> 0,0000</td>
<td>93,0000</td>
<td>0,0000</td>
</tr>
<tr>
<td></td>
<td><strong>Avg.</strong> 0,0021</td>
<td>93,0000</td>
<td>0,0287</td>
</tr>
<tr>
<td></td>
<td><strong>Max.</strong> 0,0081</td>
<td>93,0000</td>
<td>0,0650</td>
</tr>
<tr>
<td><strong>128</strong></td>
<td><strong>Min.</strong> 0,0000</td>
<td>93,0000</td>
<td>0,0000</td>
</tr>
<tr>
<td></td>
<td><strong>Avg.</strong> 0,0049</td>
<td>93,0000</td>
<td>0,0474</td>
</tr>
<tr>
<td></td>
<td><strong>Max.</strong> 0,0120</td>
<td>93,0000</td>
<td>0,0690</td>
</tr>
<tr>
<td><strong>256</strong></td>
<td><strong>Min.</strong> 0,0043</td>
<td>94,0000</td>
<td>0,0610</td>
</tr>
<tr>
<td></td>
<td><strong>Avg.</strong> 0,0256</td>
<td>94,0000</td>
<td>0,0644</td>
</tr>
<tr>
<td></td>
<td><strong>Max.</strong> 0,0610</td>
<td>94,0000</td>
<td>0,0680</td>
</tr>
<tr>
<td><strong>512</strong></td>
<td><strong>Min.</strong> 0,0012</td>
<td>93,0000</td>
<td>0,0260</td>
</tr>
<tr>
<td></td>
<td><strong>Avg.</strong> 0,0329</td>
<td>93,7143</td>
<td>0,0845</td>
</tr>
<tr>
<td></td>
<td><strong>Max.</strong> 0,1300</td>
<td>94,0000</td>
<td>0,1400</td>
</tr>
<tr>
<td><strong>1024</strong></td>
<td><strong>Min.</strong> 0,0250</td>
<td>93,0000</td>
<td>0,0440</td>
</tr>
<tr>
<td></td>
<td><strong>Avg.</strong> 0,0750</td>
<td>94,0000</td>
<td>0,1300</td>
</tr>
<tr>
<td></td>
<td><strong>Max.</strong> 0,0750</td>
<td>94,0000</td>
<td>0,1300</td>
</tr>
<tr>
<td><strong>1280</strong></td>
<td><strong>Min.</strong> 0,0000</td>
<td>92,0000</td>
<td>0,0000</td>
</tr>
<tr>
<td></td>
<td><strong>Avg.</strong> 0,0018</td>
<td>93,1429</td>
<td>0,0300</td>
</tr>
<tr>
<td></td>
<td><strong>Max.</strong> 0,0314</td>
<td>93,0000</td>
<td>0,1300</td>
</tr>
<tr>
<td><strong>1518</strong></td>
<td><strong>Min.</strong> 9,5000</td>
<td>93,0000</td>
<td>22,0000</td>
</tr>
<tr>
<td></td>
<td><strong>Avg.</strong> 9,9100</td>
<td>93,4286</td>
<td>22,2000</td>
</tr>
<tr>
<td></td>
<td><strong>Max.</strong> 10,0000</td>
<td>94,0000</td>
<td>23,0000</td>
</tr>
</tbody>
</table>