Erkki Hurme

Evaluation of SLA Monitoring System

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Thesis

8.2.2016

Tiivistelmä

Tekijä Otsikko Sivumäärä Aika	Erkki Hurme SLA valvontajärjestelmän arviointi 32 sivua 8.2.2016
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Tietotekniikka
Suuntautumisvaihtoehto	Tietoverkot
Ohjaaja	Lehtori Marko Uusitalo

Tämän työn tarkoitus oli arvioida Creanordin teleoperaattoritason SLA hallinnointiohjelmisto Echovaulttia. Echovaultin iso valtti on sen tuki useiden eri valmistajien kytkimille ja reitittimille. Työssä haluttiin testata TWAMP-protokollaa Creanordin Creanode 3000 verkkotestilaitteen ja Metropolian verkossa jo olevien eri laitteiden välillä.

Testaus toteutettiin Metropolian kampusverkossa käyttäen Metropolian laitteita. Ensimmäiset testit tehtiin kahdella ethernet NID:llä, jonka jälkeen Creanode 3000 ja Cisco ME-C3750 Ethernet swichin välinen TWAMP testaus aloitettiin. Echovaut palvelinta käytettiin hallinnoimaan kaikkia laitteita. Testien tarkoitus oli selvittää laitteistojen yhteensopivuus ja arvioida niiden toimivuutta. Käytännössä testit toteutettiin asettamalla laitteiden asetukset ja testit Echovaultin web-käyttöliittymän kautta. Osan laitteiston asetukset täytyi asettaa manuaalisesti, koska ne eivät tukeneet integrointia Echovaultin kanssa.

Testeissä kävi ilmi, että Creanode 3000 ja Cisco ME-C3750 välisissä TWAMP testeissä on ongelmia, jotka mahdollisesti johtuivat puutteellisesta TWAMP tuesta. Työn aikana ongelman lähdettä ei pystytty varmistamaan, koska Metropolialta ei löytynyt operaattoritason laitteistoa, joka täysin tukisivat TWAMP:ia. Tämän ongelman jatkoselvittäminen voisi olla aihe jatkotutkimukselle. Työn aikana myös todettiin, että Echovault on monimutkainen, mutta tehokas työkalu hallinnoimaan erilaisia verkkotestejä.

Avainsanat	SLA, TWAP, Creanode, Echovault

Abstract

Author(s) Title Number of Pages Date	Erkki Hurme Evaluation of SLA monitoring system 32 pages 8 February 2016
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialization option	Data Networks
Instructor(s)	Marko Uusitalo, Senior Lecturer

The purpose of the thesis was to evaluate Creanord's carrier-grade multi-platform SLA management software Echovault and test its TWAMP functionality with Creanord's Creanode 3000 network measurement device and other equipment provided by Metropolia.

The testing was done using Metropolia's campus network, Ethernet NIDs, Creanode 3000, Cisco ME-C3750 Ethernet switch and Echovault. In the tests the functionality and compatibility of the devices and software were tested and evaluated. The testing was done by configuring the Creanode 3000 and NIDs through Echovault and manually configuring the other devices.

In the tests it was found that there is a problem using TWAMP between Creanode 3000 and Cisco ME-C3750 switch with the 12.2(58)SE2 firmware version. Figuring out where the problem was would be a good subject for future research. Echovault was also evaluated to be complex and powerful but at first confusing multi-platform SLA management software.

Keywords	SLA, TWAMP, Ping, Cisco

Table of Contents

1 Introduction

Internet and high connectivity is a considerable part of almost any business nowadays. Companies and businesses relay heavily on properly functioning internet and interconnectivity between offices that are geographically apart. This connectivity is provided to the companies by service providers. The service providers usually have contracts with internet operators which then have contracts with bigger operators that provide access to an even bigger operator's network, and in the end when all of this sums up we have what we call the internet. Service providers sell connections that are highly complex to the customers. To assure that the systems function on a level that is desired by the customer and that the customer gets what they pay for some rules must be agreed on. For this purpose service-level agreements (SLAs) were created.

To be able to provide services meeting the requirements of the SLAs, service providers and internet operators need a standardized way to measure SLA's between networks that are controlled by different entities. Operations, administration and maintenance (OAM) tools and standards have been developed for this purpose. OAM provides a standardized way to efficiently monitor, troubleshoot, manage and monitor performance of a network by using standards such as ITU-T Y.1731, IEEE 802.12ab, IEEE 802.1ab, IEEE 802.3ah and TWAMP.

In the theory part of the study the different standards and their roles are covered. In the practical part it is explained how two NID's were installed, configured and attached to service level agreement (SLA) management software called Echovault. After getting started with the Echovault device called Creanode 3000 (CN3K) was used. CN3K is a product from company called Creanord. CN3K is a multi-vendor performance measurement probe that uses open measurement standards, it is used with Cisco's Metro Ethernet switch ME-C3750 to test CN3K's and Echovault's TWAMP-functionality.

2 TWAMP, Echo and SLA

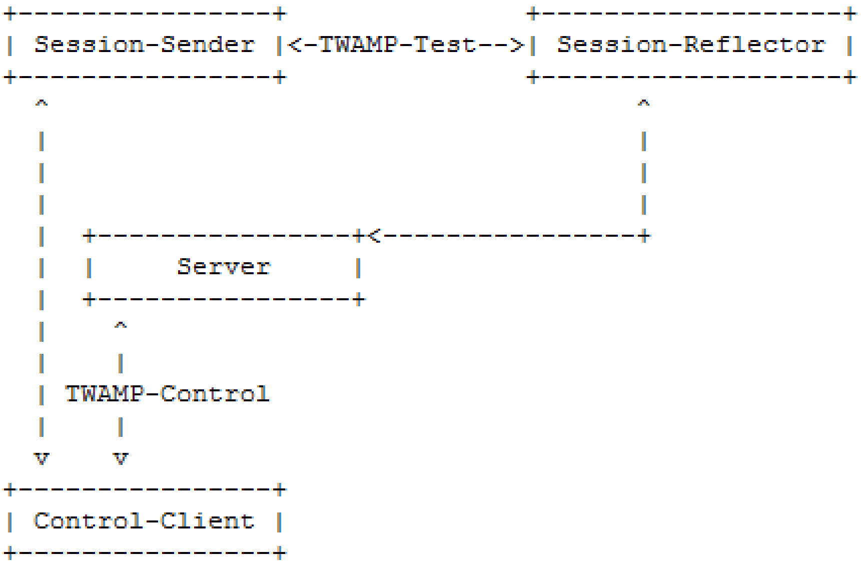
TWAMP and UDP Echo are technologies used to confirm whether host is available or not. In principle both of them can be used check host availability, however practicality, security, accuracy and functionality wise the TWAMP outclasses the Echo Protocol in every possible way. Echo Protocol could be called old relic from bygone era, but it was one of the first ways to confirm host availability around the time it was standardized. However in modern networks there is no space for it, and protocols like TWAMP are needed to uphold Service Level Agreements (SLA).

2.1 TWAMP

Two-Way Active Measurement Protocol (TWAMP) is an active network measurement protocol which was first defined by the Internet Engineering Task Force (IETF) in Request for Comments (RFC) number 5357. The benefits of using active measurement instead of passive is elevated privacy and accuracy at the cost of network bandwidth that the measurements generate and higher CPU and ram utilization, which has lately become less of an issue due to faster hardware and higher capacity ram. [1, p. 1] Round-trip measurements or in other words two-way measurements are widely used in network solutions, because they do not need clock synchronization at the remote node.

The most widely known example of two-way measurement is ping which uses (ICMP Echo Request/Reply) to calculate round trip time (RTT). Ping however has few weaknesses, because it relays on ICMP. ICMP packets are handled differently between different platforms which causes additional variation in measurements. This variation is not present if TCP or UDP is used, because TCP and UDP protocols are handled more uniformly between different platforms. ICMP packets are also handled differently in comparison to TCP and UDP packets by routers. In a case of high network load it is not uncommon for a router to start dropping ICMP packets, while still forwarding TCP and UDP packets normally. [2]

TWAMP is based on One-Way active measurement Protocol (OWAMP), and it adds two-way (round-trip capabilities) to it. TWAMP uses well-known TCP/UDP port 862. TWAMP protocol logical model consists of 4 parts (see Figure 1 below). [3, p. 5]

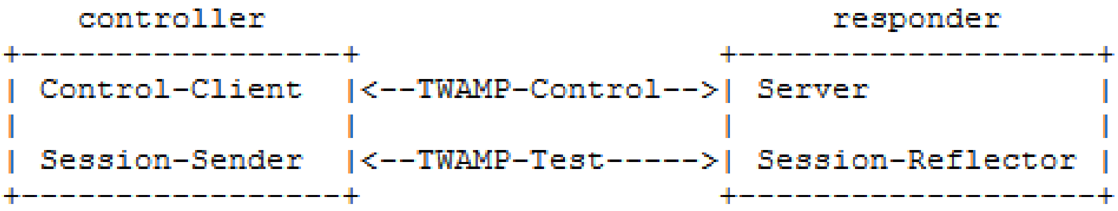


```
+----------------+               +-------------------+
| Session-Sender |<-TWAMP-Test-->| Session-Reflector |
+----------------+               +-------------------+
  ^                                     ^
  |                                     |
  |                                     |
  |                                     |
  |  +----------------+<----------------+
  |  |     Server     |
  |  +----------------+
  |    ^
  |    |
  | TWAMP-Control
  |    |
  v    v
+----------------+
| Control-Client |
+----------------+
```

Figure 1. TWAMP logical model [3, p. 3]

Control-Client initiates and terminates TWAMP-test sessions. It does it by using TWAMP-Control protocol to make the Server activate the Session-Reflector. After the Session-Reflector is activated the measurement packages sent by Session-Sender can be received and replied to. [3];[4]6;7

In TWAMP more than one of the roles can be hosted by the same host. In the RFC 5357 one example of how to divide the roles is given. The example divides the roles to two hosts. One of the hosts is both Control-Client and Session-Sender and the other one is Server and Session-Reflector as shown in Figure 2 below. [3, p. 3]



```
    controller                                 responder
+-----------------+                   +-------------------+
| Control-Client  |<--TWAMP-Control-->| Server            |
|                 |                   |                   |
| Session-Sender  |<--TWAMP-Test----->| Session-Reflector |
+-----------------+                   +-------------------+
```

Figure 2. Example from RFC5357 which shows how the roles could be divided. [3, p. 3]

TWAMP standard does not specify how often the test packets are sent in a test session, the send interval is specific to implementation [3, p 12]. The format of the unauthenticated and unencrypted test packet sent by Session-Sender is shown in Figure 3. Each packet will have their unique sequence number to identify test packets from each other. The timestamp consists of two parts. The first part tells how many full seconds has passed since 00:00 on 1st of January 1900 at the time of sending of the packet (Unix time). The second part of the timestamp tells the remaining fractional part of the second of the sending time. The error estimate part of the packet includes the information of how accurate the clock synchronization is and if it uses NTP server or GPS hardware for clock synchronization. Packet padding is used to make the packet same size as the packet later returned by the session-reflector, this is needed because the packet returned has more fields than the packet sent by the session-sender.

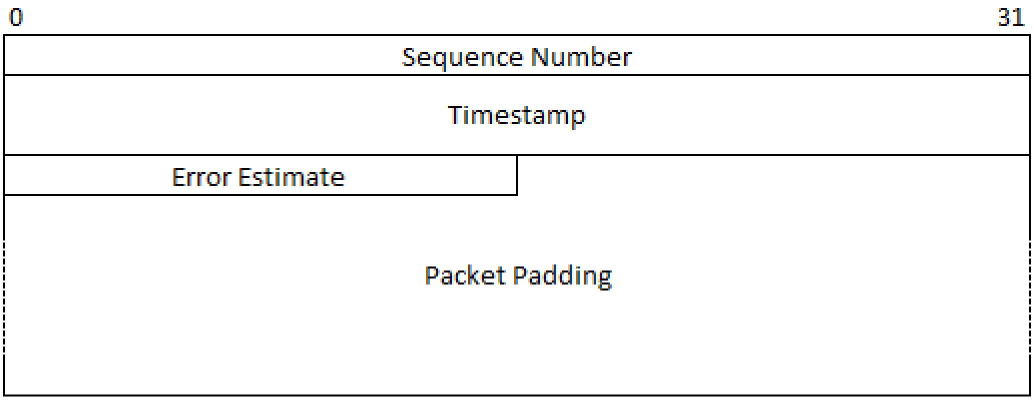


Figure 3. TWAMP test packet format, session-sender [5, p 29]

The test packet returned by the session-reflector after receiving the test-packet from session-sender can be seen in Figure 4. Session-reflector returns the packet to session sender as quickly as possible. The packet sent by session-reflector has all the same parts as the packet sent by session-sender, and some additional fields. The first new field in the packet is Must Be Zero (MBZ) which will be set to zero and it will be ignored by the receivers. Receive timestamp is the time when session-sender received the test packet. Sender sequence number is independent from the sequence number in the received packet. Sender timestamp and sender error estimate have the same format as the in the packet received from the sessions-sender, but they have the information from the session-reflector. Sender TTL will be the time to live from the header of the received IP packet. The packet padding will depend on the size of the received packet. The packet size of the new packet must match the size of the packet that was received. [3, p. 12]

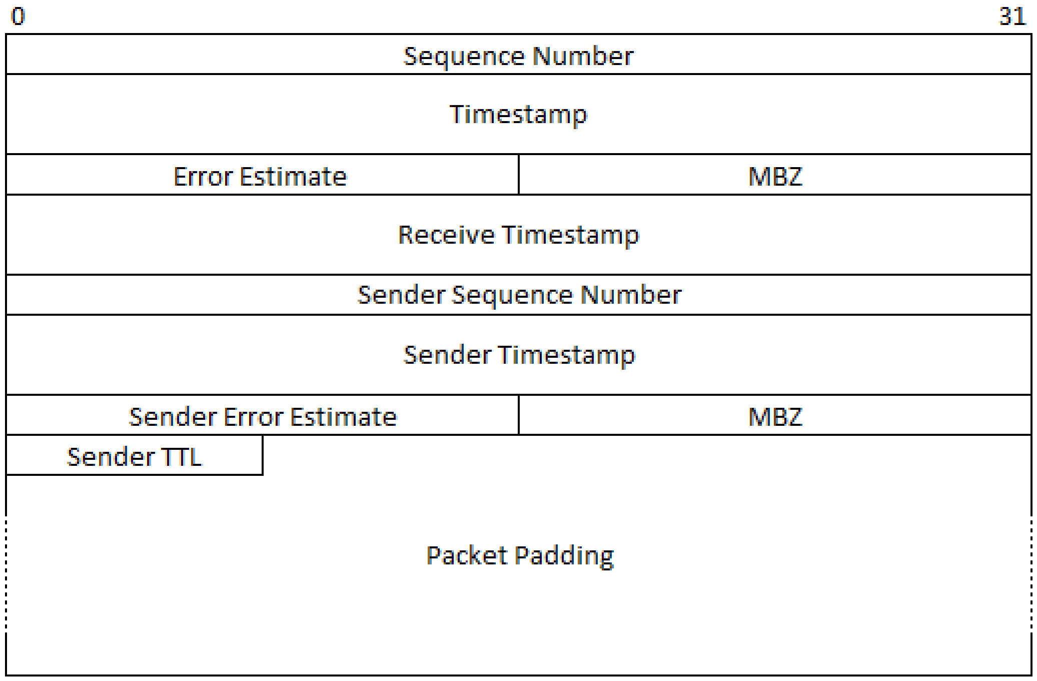
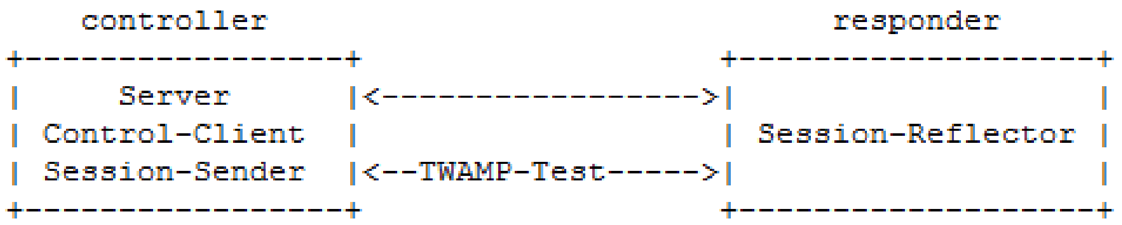


Figure 4. TWAMP test packet format, session-reflector [3, p. 15]

The TWAMP-test protocol can be used in three different modes. These three modes are unauthenticated, authenticated and encrypted. The Figures 3 and 4 show the packet format for unauthenticated packets. Authenticated and encrypted test packets have additional fields to enable the added functionality.

2.2 TWAMP Light

The Internet Engineering Task Force (IETF) also makes a note of a lighter version of TWAMP called TWAMP Light, in which server role is also moved to the same host as control-client and session-sender. This host is called controller in this case, and leaves only the Session-Reflector role on the other host which is called responder. In this approach the responder can be a simpler node which only reflects the test-packages sent by the controller. Figure 5 illustrates the TWAMP Light role model.



```
     controller                                 responder
+------------------+                    +--------------------+
|      Server      |<------------------>|                    |
|  Control-Client  |                    |  Session-Reflector |
|  Session-Sender  |<--TWAMP-Test------>|                    |
+------------------+                    +--------------------+
```

Figure 5. TWAMP-light role model. [3, p. 22]

This approach removes the need for TWAMP-Control protocol by assuming that the responder is configured by non-standard means. In a case of TWAMP Light it is possible that the responder does not know the state of the TWAMP session, because it is so by design and the only job of the responder in this case is to copy the Sequence Number from the received package and paste it to packet to be reflected while also generating the necessary timestamps for one way delays. By removing the need for TWAMP-control, TWAMP Light becomes easier for the hardware and software vendors to implement and support in their operating systems and devices. [3, p. 22]

2.3 Echo Protocol

Echo Protocol which is a part of the Internet protocol suite, that can use TDP or UDP connection. The echo protocol uses the well known port number 7. In echo protocol the client establishes a connection to the server and the server simply sends back all the information it receives from the client until the client closes the session. [4] Echo Protocol can be used for debugging and confirming if the host is answering or not. Echo Protocol was first introduced in RFC 862, which dates back to May 1983. Any modern application should not be relaying on it. If Echo Protocol is enabled, attacker can use IP spoofing to make the Echo Protocol enabled host to attack any reachable IP address.

2.4 SLA

In modern day high availability and reliability are key points to commercial networks. Short outages in a network can get expensive really quickly, for example in a case of network outage at a bank even a few seconds can become really expensive. Service-Level Agreement (SLA) is a solution in which the customer and service provider agree to a set of parameters which the service provider guarantees to deliver. These parameters include but are not limited to network availability, latency, jitter, packet loss and throughput.

In this case network availability means the time that the service is accessible and working for the customer. In computer networks the time that a packet takes to travel from host A to host B is called latency and the variation of latency between consecutive packets is called jitter. Latency can be one-way or round trip. Network throughput is the total bandwidth available for the customers. These are just the most used parameters for SLAs, there are many other parameters that can be measured but there is also parameters that can not be measured so accurately and easily. An example of the later would be the modifiability of the network, and how often a service is likely to change. Security of a network is also hard if not impossible to measure conclusively. [6, p. 16]

An enterprise customer can choose the level of quality of service it needs to run its business and the service provider must assure that this level is met. SLA also usually describes who makes the measurements and how. Sanctions for not being able to meet the agreed level is also described. This kind of service is usually not available for private customers. It is mainly meant for companies which businesses rely heavily on the internet services. For companies even small outage can mean big losses. When the company’s bottom line relays on internet service functioning properly, companies are much more willing to pay for a service which assures that the service will function as promised. Also if one wants to get SLA with higher requirements and stricter limits, one has to pay considerably more.

3 Operations, Administration and Management (OAM)

Operating and managing any kind of advanced system is a complicated task. To ease the operating of these kinds of systems, different tools and standards have been and are being devised. These tools and standards in combination form the Operations, administration and management (OAM). In the operator’s network these tools can be used for variety of things, for example monitoring networks and servers, testing the performance of the connection to see if it meets the SLA guidelines, detecting and locating problems and keeping track of the customer’s service usage. TWAMP in particular can be used for both monitoring connections and testing SLAs. Here are some other standards that are part of the OAM.

- ITU-T Y,1731 Performance Monitoring
- IEEE 802.1ab Link Layer Discovery Protocol
- IEEE 802.1ag Connectivity Fault Management (CFM)
- IEEE 802.1ah Transport Layer OAM Link-Fault Management

OAM tools can be classified into three different layers depending on which layer of OSI model they operate on. These layers are service layer, connectivity/network layer and link/transport layer. [7; 8, p. 3]

3.1 ITU-T Y.1731 and IEEE 802.1ab and IEEE 802.3ah

ITU-T Y.1731 PM, provides means to monitor performance of a service to see if it can comply with the SLA agreed between service provider and the customer. Y.1731 PM can measure three different aspects of a networks performance. These are frame loss, frame delay and delay variation. [9, p. 10]

The main purpose of the 802.1ab is to provide network administrators a standardized way to discover devices, device failures and problems in configurations in a multi-vendor network. IEEE 802.1ab does this by defining the LLDP protocol. The link Layer Discovery Protocol (LLDP) is an open discovery protocol for local and metropolitan networks. It provides similar functionality as Cisco Discovery Protocol (CDP) and other similar proprietary solutions. LLDP works over Ethernet network and allows compliant devices to discover neighboring devices, their capabilities and advertise their own connectivity and capabilities to their neighboring devices. [10]

EEE 802.3ah transport layer OAM link-fault management focuses on the first or last mile infrastructure from the customer equipment to the service provider’s network. It is only a single hop/wire protocol and it is not aware of the rest of the network or service. Its main function is OAM discovery. It helps to discovery the level of OAM support the device is capable of. For example it can discover if the device is capable of dying gasp which employs big enough capacitors or some other external power source, so when a power failure occurs it can send an error message before shutting down, fault isolation that can isolate whether the problem is in the customer’s or the providers network and if there is an unidirectional failure. 802.3ah also supports port level loopback, which helps to troubleshoot connection by setting a remote node to state where it reflects all inbound traffic back on the same link. [11, p. 13]

3.2 IEEE 802.1ag Connectivity Fault Management (CFM)

IEEE 802.1ag is a standard to help discover, verify and isolate connectivity problems in a network which is operated by multiple different organizations, which each have their own equipment and limited or no access to each other's equipment. IEEE standard 802.1ag about Connectivity fault Management (CFM) consists of two main parts. The first part defines logical parts of a network. These logical parts are maintenance domains and maintenance points.

Maintenance domains (MD) are one part of network that one operator or one service provider is in charge of. The main point of domains is to have clear understanding who is in charge of which equipment and what part of a network in a case of network problem, so the one responsible can be identified quickly. Maintenance domains can nest but can not overlap. Overlapping would mean that more than one entity would be in charge of same part of network and which is not allowed. Maintenance domains are divided to eight levels. Higher domain level corresponds to bigger domain. Domain levels go from 0 to 7. Where 7 is the biggest level available. Level 7 domain usually covers the whole service from the customer's network to their other network. Operators' domains are usually level 0 and service providers are from between 1 and 6. [12, p. 3; 13] Figure 6 illustrates the maintenance domains.

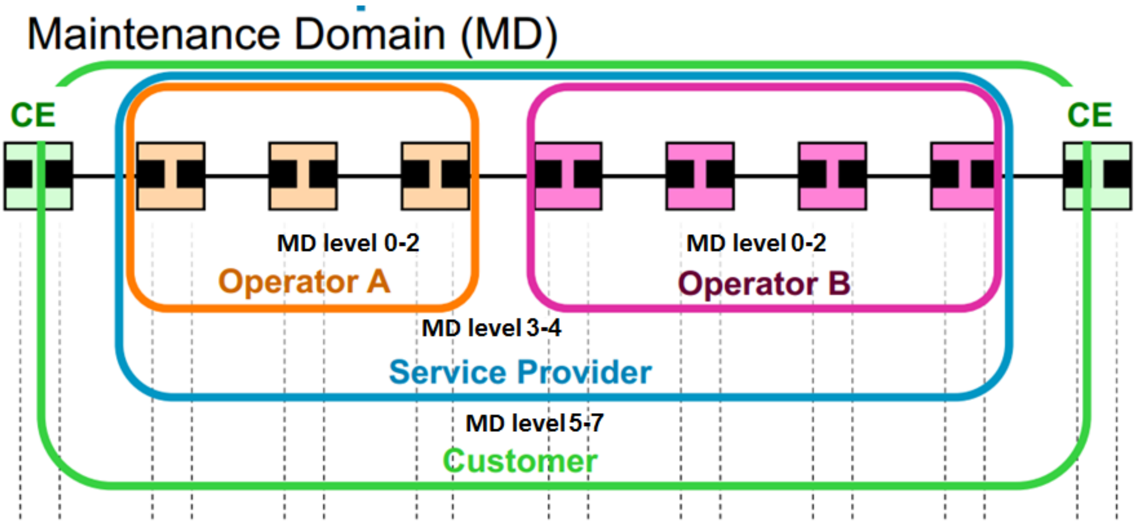


Figure 6. Maintenance Domains [11, p. 19]

In CFM two kinds of logical part of MDs are defined Maintenance Association End Points (MEP) and Maintenance Domain Intermediate Points (MIP). MEPs are at the edges of a network and define the domain. MEPs drop all CFM frames with lower or same MD level than their own, that come from outside of the domain and process frames of a same MD level that come from inside the domain and drop the frames with lower MD level than their own. MEPs forward all frames from higher level MDs regardless of where they come. MIPs are inside domains and not at the edges. They forward frames with same or higher MD level and drop frames with lower level. [12, p. 7]

Inside maintenance domains the connectivity of a service is monitored. The services are monitored between MEPs. In the following example the service is monitored on 4 different levels.

- Customer to Customer
- Operator A outer edge to Operator B outer edge (Service Provider's network)
- Operator A outer edge to inner edge (Operator A's network)
- Operator B outer edge to inner edge (Operator B's network)

Failure between any MEPs can be detected and this hierarchy helps in locating the failure point whether it is in the customer's, service provider's or operator's network. Figure 7 illustrates the CFM model.

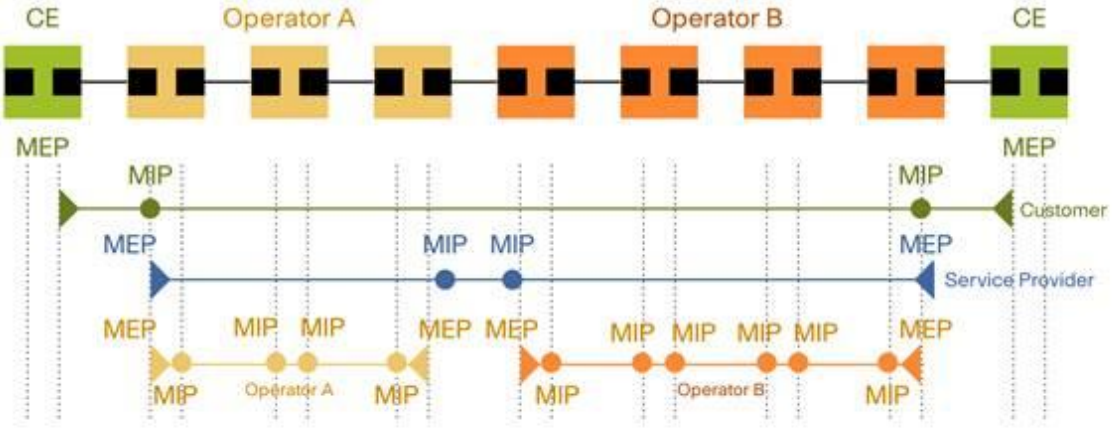


Figure 7. CFM model [11, p. 22]

The managed objects which are necessary to manage all of this are also defined. By clearly defining administrative areas and who is in charge of what equipment makes it easier to act when something goes wrong [14]. Protocols part of CFM employ regular Ethernet frames that are sent over the same volume as normal traffic. Devices that do not support CFM simply forward the packets as normal Ethernet packets. [8]

The second part of the 802.1ag CFM defines three protocols for the maintenance points to help discover, verify, isolate and report Ethernet connectivity problems and to maintain operational networks [14]. These three defined protocols are Continuity Check Protocol (CCP), Loopback Protocol and Linktrace Protocol.

CCP sends continuity check messages (CCMs) which are multicast heartbeat messages sent between MEPs which are edge devices in maintenance domains. These heartbeats are used to help MEPs to discover other MEPs and to help other network devices (Maintenance Intermediate Points: MIPs) to discover MEPs. CCMs are confined to a domains by MEPs. [13, p. 8]

Loopback protocol is quite similar in concept to ICMP ping. It is used to confirm if MEP or MIP can be reached by sending a unicast message to the device. By receiving the answer to the loopback message the state of the MP can be confirmed but like with ICMP ping the route/path can’t be confirmed. These messages are sent manually by the administrator from CLI.

Linktrace Protocol is similar to the loopback protocol and is requested by the administrator manually from the CLI. Linktrace protocol is similar in concept to regular IP traceroute, but it operates on the data link layer of OSI mode instead of network layer. It is used to discover all MIPs on a path to a specific MEP in a same maintenance domain by sending Link-trace Message (LTM) which use multicast frames. Each MIP and in the end the destination MEP each reply to the LTM with a unicast Linktrace Reply (LTR) and forward the LTM to the next hop. [13, p. 8]

4 Artificial Manipulation of Network Functionality

To test the accuracy of a network device and to test how they react to network not working optimally, correctly or is simply low on bandwidth, a way to simulate different kinds of networks and network loads is necessary. For that purpose a computer with Linux operating system with two networks interfaces can be used. The computer can be used as a router between the test devices. By having the Linux based router between the devices, variety of variables can be controlled, and by changing these variables, variety of different types of networks can be simulated. These variables are as follows.

- Packet loss, one or more packets get lost on the way.
- Packet corruption, set amount of bit errors in packets.
- Packet re-ordering, reorder the order that packets arrive.
- Artificial delay, delaying packets by set amount of time.
- Jitter, creating random delay on top of artificial delay.
- Rate control, artificially limit the bandwidth to create a bottleneck.

By combining the different variables above it is possible to emulate many kinds of networks. Networks with broken or failing hardware, networks that are under a heavy load, networks that have one weak point which acts as a bottleneck, networks where packets take different routes and networks which combine all of these problems, can be emulated with Linux based router by using netem and tc.[15]

To create the previously listed effects a tool called tc (traffic control) and netem (Network Emulation) can be used. The latter is included in the Linux kernel and the former is part of a tool package called iproute2. Netem is included in the most common Linux distributions, if the kernel that the distribution is using is newer than 2.6. In the present study the Debian stable version 7.5 was used. The Debian version 7.5 uses the version 3.2 of the Linux kernel and so supports netem by default. [15]

To configure netem a program called tc can be used. Tc is a program that can be used to show and manipulate network traffic control settings. With tc netem rules can be added to queuing discipline (qdisc) of a specified network interface or device. Qdisc is a buffer between kernel and the network driver. The kernel first places packets to the qdisc queue configured for the specific interface and then immediately tries to pass as many packets as possible to the network driver. With netem it is possible to manipulate this process. [16]

A simple example of this would be adding 1s delay to network device called eth0. The command below only affects outgoing traffic. It is possible to use tc and netem to control also incoming traffic but for this project the ability to control just outgoing traffic is enough, because it is done in both directions. The Linux router is between the test devices and by manipulating the two interfaces on the computer it is possible to control both incoming and outgoing traffic.

```
tc qdisc add dev eth0 root netem delay <delay> (<jitter >)
(<correlation in %>)
```

Example code 1. Adding delay to an interface with tc.

The only required parameter when adding delay is the desired delay in milliseconds. Adding random jitter and correlation between consecutive packages is optional. Packet loss can be manipulated in almost the same manner. The packet loss percentage is the only required parameter for packet loss when using netem. The percentage correlation between consecutive packages is again optional. In a case of a packet loss, correlation is useful because it can be used to emulate bursts of packet loss. [15]

```
tc qdisc change dev eth0 root  netem loss <packet loss in %> (<correlation>)
```

Example code 2. Causing packet loss to an interface with tc.

Bursts of packets getting lost can occur in many different ways. For example route can become overloaded, hardware might fail or there can be temporary interference while using wireless connection. Reasons for packet loss are endless when talking about system as wide as the internet.

4.1 Network Shaping and Token Bucket Filter

In test environments the network is usually simple and the bandwidth between test hosts is usually the maximum speed of the interfaces. For example 100 Mbit/s or 1 Gbit/s. In some network testing scenarios one may however wish to limit the bandwidth to something similar to a DSL line or something similar with lower bandwidth. This can be done with network shaping and token bucket filtering. Usually this would be used to move the weakest link of a network, or in other words the bottleneck of the network to a desired location. By doing this, the ability to control the network will increase and the network will be more predictable. Network shaping on a Linux machine can be done using queuing disciplines.

Token Bucket Filter (TBF) is a classless queuing discipline that can be used with tc. This only works for outgoing packets, but since the Linux machine can be placed between the test devices, the traffic can be controlled in both ways. TBF was chosen because it is best suited for the project at hand, since in this case it is only wanted to limit the bandwidth of the interfaces. Packet prioritization and such which are the biggest benefits of a more advanced classful queuing disciplines are not of interest here.

> TBF is very precise, network- and processor friendly. It should be your first choice if you simply want to slow an interface down. [17]

In TBF there is a virtual bucket that contain tokens. Those tokens are replenished periodically at a configured rate. Every packet that is sent consumes a token from the bucket. If the traffic is slow enough, not to cause the bucket to get empty, the packets are sent as they get queued. If the bucket runs out of tokens the packets will wait for new tokens. If enough tokens will not arrive in configured time (latency) the packet will be dropped. [19]

```
tc qdisc add dev eth0 root tbf rate <desired bandwidth>
burst <burst rate> latency <latency> mtu <mtu of the inter-
face>
```

Example code 2. Limiting bandwidth of an interface with tc.

The burst size has to be at least the desired bandwidth divided by the kernel's interrupt frequency in the system. In this case the frequency was 250 Hz. To achieve 5 mbps connection, the burst size of at least 2 500 bytes is necessary as calculated below. [18]

(5*10^6) / 250 = 20 000 bits = 2 500 bytes

The MTU of the Token Bucket Filter also needs to be same as the MTU of the interface. If the MTU do not match, TBF won’t function properly. Token Bucket Filter is a simple but robust classless queuing discipline which can do simple network shaping.

5 Creanord, Echovault and Hardware

Echovault is a software created by company called Creanord. In this chapter the company, the software and compatible hardware will be explored.

5.1 Creanord and Echovault

Creanord is a Finnish company that was founded in 2000. Creanord’s office is located in Helsinki and their main focus is in multi-Vendor SLA management software and hardware. [19]

Figure 8. Creanord Logo

In the present study Creanord’s SLA management software Echovault and their ultra-performance network measurement probe Creanode 3000 were heavily used.

> Echovault is a Carrier-Grade service and SLA delivery platform for service providers, network carriers, enterprises and government IT organizations. [20, p. 2]

Above is Creanord’s short answer to the question what Echovault is. In other words Echovault is multi-platform SLA management software, which works in unison with many kinds of network equipment from different manufacturers. Creanord also designs and manufactures its own precise measurement devices, which can be used to add functionality and additional measurement capabilities to Echovault. Figure 9 shows the login page to Echovault.

Figure 9. Echovault login

Echovault is completely configured and monitored through a webpage, which can only be accessed after authentication with username and password. Echovault provides groups to help manage users and the level of access each of them are allowed to have. Customers can be limited to only see the reports or parts of the reports that they need to have access to, and if necessary hide all the numerous other settings and reports that they do not need access to.

Echovault with all its settings can be quite overwhelming to digest, especially if one is not familiar with the relevant technologies and terminology. Below the logical model of Echovault is summarized. The explanation starts from the bottom and proceeds from smaller to bigger. First there are test devices called nodes. To configure the nodes to perform tests and measurements one first needs to include the node in a policy or an advanced policy, depending on a type of a node or device in question. Policies are then part of spotlights, which can be configured to be part of dashboards. Echovault then also has SLA engines and SLA profiles which are used to compute the data in to results. Configuring SLA thresholds, other parameters and how often the data is calculated in to results is also configure inside SLA engines and SLA profiles. Figure 10 illustrates the different parts of Echovault.

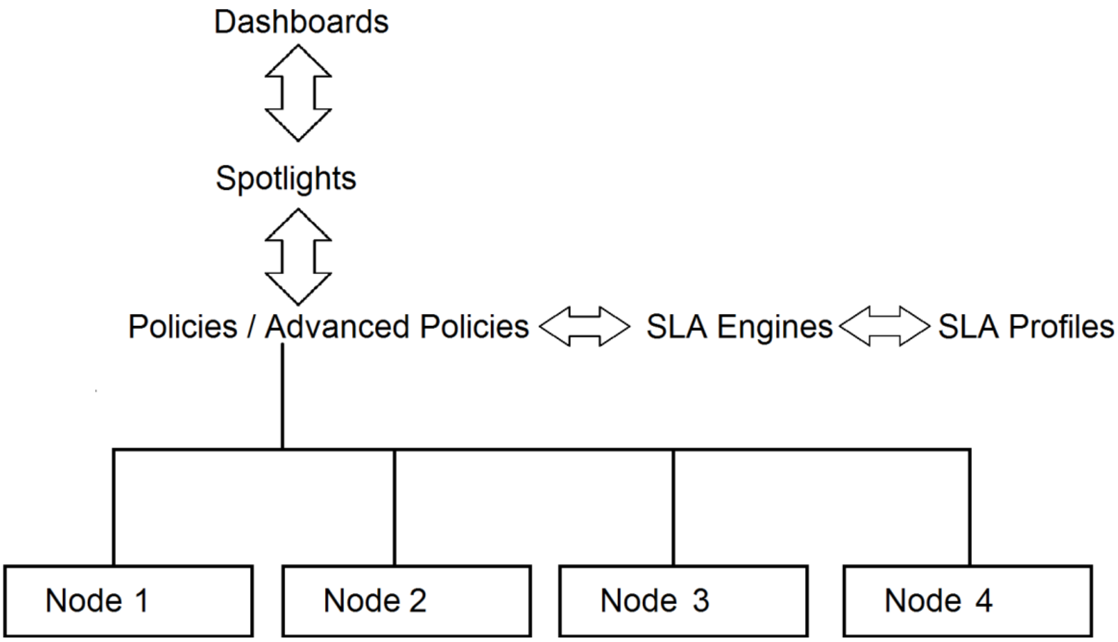
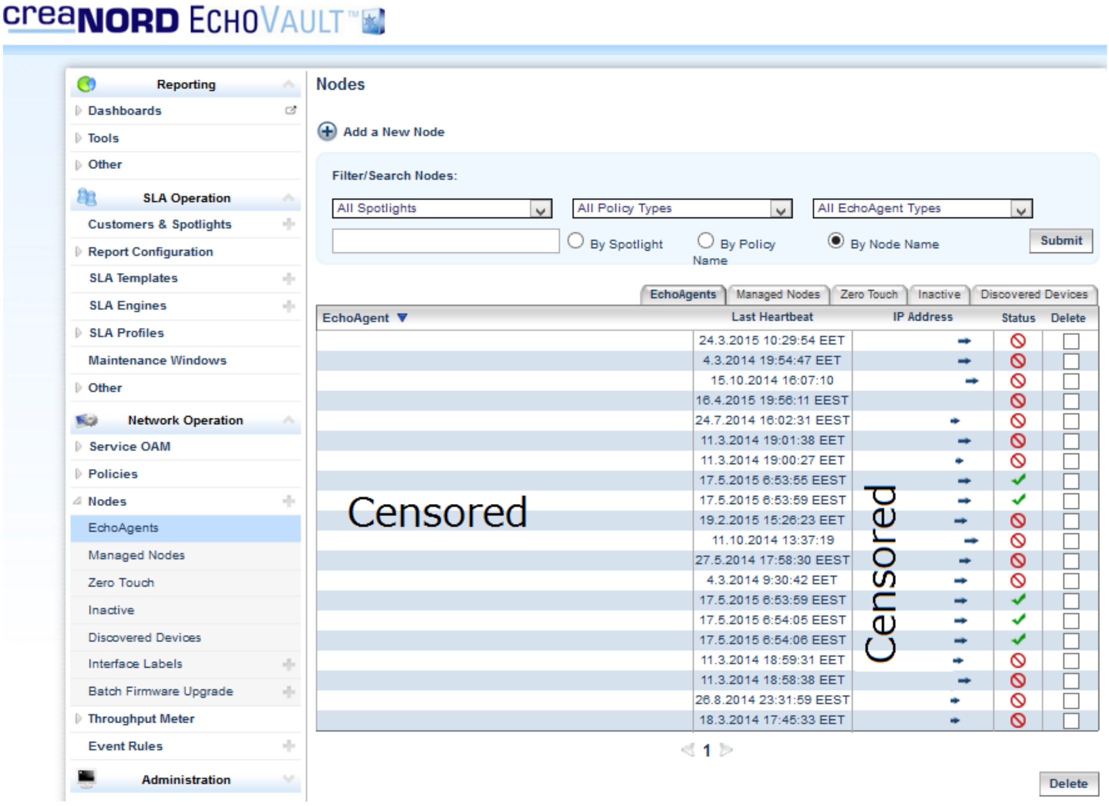


Figure 10. Echovault parts

In the management webpage Echovault divides everything under four main titles. Reporting, SLA Operation, Network Operation and Administration, as shown in Figure 11.



Nodes

Add a New Node

Filter/Search Nodes:

All Spotlights | All Policy Types | All EchoAgent Types

○ By Spotlight ○ By Policy Name ● By Node Name — Submit

EchoAgents | Managed Nodes | Zero Touch | Inactive | Discovered Devices

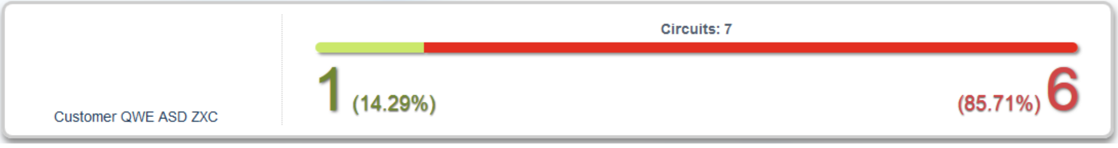
EchoAgent ▼	Last Heartbeat	IP Address	Status	Delete
Censored	24.3.2015 10:29:54 EET	Censored ➡	⊘	□
	4.3.2014 19:54:47 EET	➡	⊘	□
	15.10.2014 16:07:10	➡	⊘	□
	16.4.2015 19:56:11 EEST		⊘	□
	24.7.2014 16:02:31 EEST	➡	⊘	□
	11.3.2014 19:01:38 EET	➡	⊘	□
	11.3.2014 19:00:27 EET	➡	⊘	□
	17.5.2015 6:53:55 EEST	➡	✓	□
	17.5.2015 6:53:59 EEST	➡	✓	□
	19.2.2015 15:26:23 EET	➡	⊘	□
	11.10.2014 13:37:19	➡	⊘	□
	27.5.2014 17:58:30 EEST	➡	⊘	□
	4.3.2014 9:30:42 EET	➡	⊘	□
	17.5.2015 6:53:59 EEST	➡	✓	□
	17.5.2015 6:54:05 EEST	➡	✓	□
	17.5.2015 6:54:06 EEST	➡	✓	□
	11.3.2014 18:59:31 EET	➡	⊘	□
	11.3.2014 18:58:38 EET	➡	⊘	□
	26.8.2014 23:31:59 EEST	➡	⊘	□
	18.3.2014 17:45:33 EET	➡	⊘	□

◁ 1 ▷

Delete

Figure 11. Echovault default front page.

Reporting as one would expect gives different ways to check the status of all of the SLAs. One quick way to do this is by using “SLA Dashboards” under Dashboard menu. “SLA Dashboards” gives an easy to understand status bar and a percentage of working circuits in each dashboard. (see figure 12)



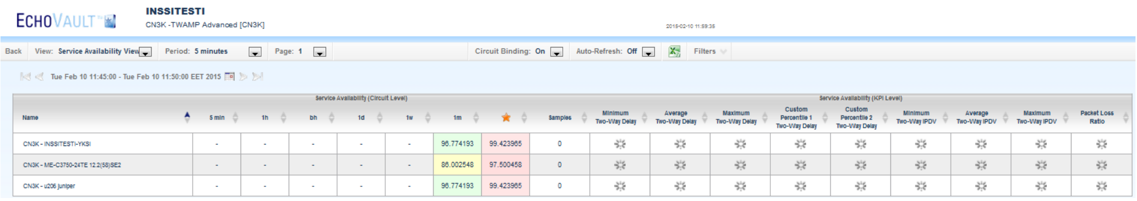
Customer QWE ASD ZXC

Circuits: 7

1 (14.29%) (85.71%) 6

Figure 12. Dashboard

By clicking one of the dashboards on the SLA dashboards, a page with a more detailed list of every policy or advanced policy is shown. The more informative Service Monitor which shows all the spotlights and their engines can be accessed from the Dashboard sub menu. From Service Monitor all the engines in different spotlights can be seen individually, even the ones that are not part of any dashboard. Information relating to the status of the services is shown here. Service availability in percentage over one hour, one day and one minute is shown first. By clicking the “analytics” on one of the spotlights one can see a more detailed view of the specific SLA engine. In the analytics view one can see the results and averages for all the tests that are performed for specific SLA engine, and results between all nodes that belong to that engine. After getting the Echovault configured and running this is one the most useful pages. One can see the general status of the SLA’s and the results for the test and if they have trigged any of the thresholds. Figure 13 illustrates this.



INSSITESTI
CN3K -TWAMP Advanced [CN3K]

2015-02-10 11:59:35

Tue Feb 10 11:45:00 - Tue Feb 10 11:50:00 EET 2015

		Service Availability (Circuit Level)								Service Availability (KPI Level)								
Name	5 min	1h	bh	1d	1w	1m	★	Samples	Minimum Two-Way Delay	Average Two-Way Delay	Maximum Two-Way Delay	Custom Percentile 1 Two-Way Delay	Custom Percentile 2 Two-Way Delay	Minimum Two-Way IPDV	Average Two-Way IPDV	Maximum Two-Way IPDV	Packet Loss Ratio
CN3K - INSSITESTI-YKSI	-	-	-	-	-	96.774193	99.423965	0	✳	✳	✳	✳	✳	✳	✳	✳	✳
CN3K - ME-C3750-24TE 12.2(58)SE2	-	-	-	-	-	86.002548	97.500458	0	✳	✳	✳	✳	✳	✳	✳	✳	✳
CN3K - u206 juniper	-	-	-	-	-	96.774193	99.423965	0	✳	✳	✳	✳	✳	✳	✳	✳	✳

Figure 13. Service Monitor - Specific SLA engine

In some cases one may want to have more detailed information about the tests, and for that under reporting there is menu called tools. In the tools menu there are pages where one can draw chart, table or export the data to CSV format and then download it. Figure 14 shows a screen shot of a timespan of 15.08.2014 – 21.11.2014 indicating the average lost packet count between CN3K and NID called INSSITESTI-YKSI. From the chart one can clearly see that lot of packets were lost on two different time spans and on one occasion there was slight packet loss. From this one can conclude that the connection was down on two occasions and that between 3th of October and 10 of October at least for short period of time there were problems between CN3K and INSSITESTI-YKSI.

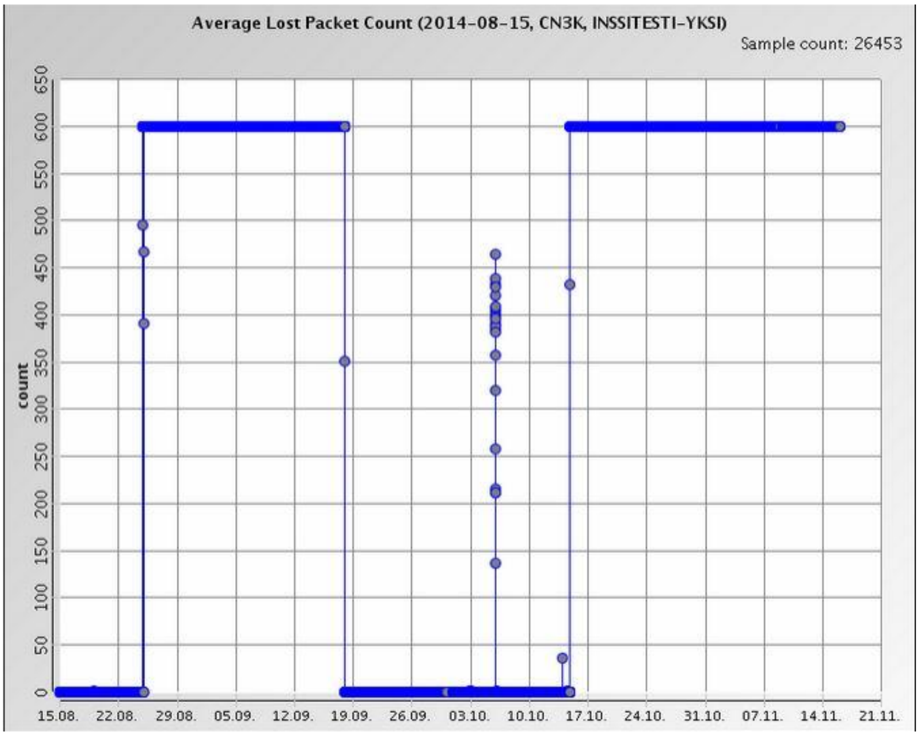


Figure 14. Echovault – Average Lost Packet Count

This kinds of charts can be useful when troubleshooting or evaluating a network. By having the baseline of a network and then seeing an increase in latency, packet loss or some other anomaly can indicate congestion or some other problem in the network that might need to be addressed sooner or later. Having the tools to gather and access this information quickly and easily can be a huge advantage, because then one can easily evaluate the networks condition and plan accordingly to minimize possible downtime from network maintenance and other problems.

SLA Operation menu includes all the settings how the test results are presented and what are the failure and warning thresholds for each test. Network Operations includes the configuration for the tests themselves and node configuration. The last menu the Administration submenu contains product information, licensees, Firmware updates, user and group management.

Overall using the Echovault can be quite confusing at first. Echovault has high configurability and support for wide range of protocols to provide the all-in-one SLA management software, however having the high configurability and support for numerous different protocols comes at a price. Getting familiar with all of the different menus and submenus can take some time.

5.2 Creanode 3000

Creanode 3000 is Creanord’s latest network measurement device, which was launched early 2015. Creanord advertises it as a game changer which simply does its own magic in already deployed multi-vendor networks. It is advertised to allow more than 10 000 concurrent test targets simultaneously, with hardware time stamping with accuracy down to 1 microsecond. [21]

CN3K supports hardware time stamping on two gigabyte Ethernet ports and two SFP’s ports. CN3K hardware includes Intel Xeon quad core, ECC memory and software on an easily switchable memory card. CN3K comes in the form factor of 1U chassis. The hardware and the software that is built on top of a Linux operating system might be able to fulfill those promises in the advertisements. However the present study only tests the CN3K TWAMP capabilities and compatibility. Performance and capabilities were not tested here. Figure 15 is Creanord’s own advertisement for the CN3K.

Figure 15. Creanode 3000 advertisement. [3]

The CN3K was delivered to Metropolia in the autumn 2014. CN3K hardware had just gotten finalized and the production had barely started. The software that was on the device was still pre-lease version and some of the debugging capabilities were still present. When first connecting to the device one was given normal Linux bash access and it was possible to poke around the system and see what exactly was inside the chassis and underneath the software. Having limited knowledge of the insides of the machine, and being able to check simple information e.g. what processor the system was running and amount of RAM et cetera was intriguing. From the Linux bash one could start the cli interface for configuring the device. On later versions this became the default behavior. Once one logged in via serial or SSH they were put straight in to the CLI for configuring the CN3K and could no longer access the Linux CLI.

The CLI of the CN3K operates similarly to the Cisco iOS devices and is easy and familiar to navigate. As on any Cisco iOS devices there are different privilege levels, e.g. the user EXEC mode, privileged EXEC mode and global configuration mode. Question mark gives all the commands available and tab-auto complete is available. In short it looks and functions like any Cisco iOS CLI, and so if one is familiar with Cisco iOS CLI, managing CN3K is a simple task.

The cli is mainly meant for only the initial configuration of the CN3K. CN3K works in close union with Echovault, so the list of things that need to be configured on the device itself is quite short.

- Configuring the IP addresses, subnet masks and default gateway, if no DHCP is available.
- Connecting the CN3K to the NTP server (optional).
- Connecting the CN3K to the Echovault server by defining its address and password.
- Configuring routes if necessary.

After setting up these three things CN3K should show up in the Echovault’s node page, and after that one is done with the initial setup. Configuring tests, monitoring and et cetera will be done through the Echovault. Wanting to change the initial settings and troubleshooting are the only few reasons why one would have to connect to CN3K directly after the initial setup.

5.3 EthernetNIDs

Ethernet Network Interface Devices (NIDs) are smaller and cheaper devices than the aforementioned CN3K. The basic idea between CN3K and NIDs is different. NIDs are usually new devices added to the network to provide additional measurement capabilities between the NIDs. CN3K on other hands focuses on leveraging the infrastructure that is already in place. NID are also cheaper low-end hardware, which means they costs less and that implementing many of them will not be too expensive, when CN3K is the polar opposite of that. CN3K is expensive high-end “all in one” solution which purpose is to take care of most of the needs with one single device, while relaying on infrastructure that’s already in place. Figure 16 is a picture of a NID.

Figure 16. NID

Usually NIDs are used in Customers premises to provide end to end measurement capability. NIDs that were accessible for this project had Echoagents installed in them, so they could also be managed with Echovault.

6 Testing

In this project the testing was done using two different protocols ping and TWAMP. Both of them have their own advantages and disadvantages. The testing was done between CN3K, NIDs and Cisco metro switch.

6.1 Ping

Even thought ping does have its flaws as a reliable test protocol, it still useful to have because of its simplicity to configure and almost nonexistent requirements. It only requires the monitored host to reply to ICMP messages which most network devices do by default. Because of this advanced ping is one of the easiest measurement to get running. In Echovault ping is called advanced ping. Figure 17 shows the advanced ping measurement results of Average Two-Way Delay between CN3K and Cisco C3750ME metro switch.

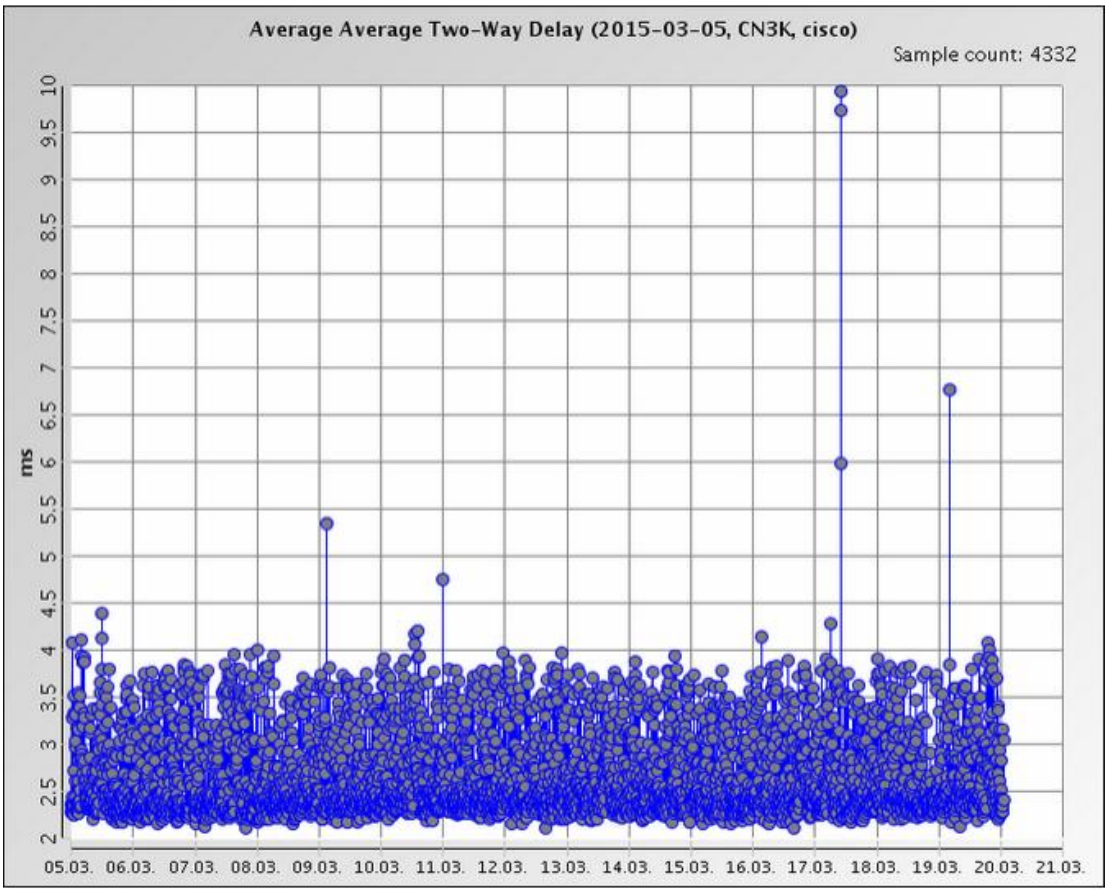


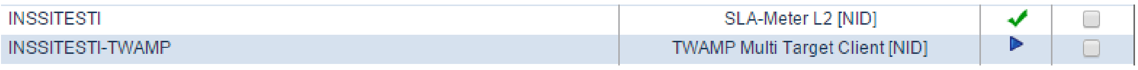
Figure 17. Average Two-Way Delay ping between CN3K and Cisco metro switch.

To configure and enable advanced ping measurements in Echovault, one has to add a new advanced policy and choose “Ping Advanded [CN3K]”. In the opening window the target hosts can be specified and other parameters adjusted. If the target device is not connected to Ethernet port 2 the port parameter must be changed in global parameters of the advanced policy.

After selecting which spotlight the measurement is part of and defining at least a single host, the measurement results can be examined with the charts tool.

6.2 TWAMP between NIDs

This thesis was started by first getting more familiar with the NIDs and Echovault in general. The first goal was to get two NIDs connected to the Echovault server. Doing this seemed like a simple task at first and it would have been over in a few days if not for a bug in Echovault. The NIDs would show up in the Echovault, but if one tried to setup any tests or change any settings of the NID from the Echovault, the settings would not get transferred to the NIDs. Echovault would just display “provisioning”, and eventually show “No Provisioning Messages received by Echovault in 5 Minutes”, Figure 18 illustrates this.



INSSITESTI	SLA-Meter L2 [NID]	✓	□
INSSITESTI-TWAMP	TWAMP Multi Target Client [NID]	▶	□

Figure 18. Echovault provisioning bug

First it was thought that the problem was with the settings of the NID or not having accurate time on the NID. However the problem still existed after resetting the NIDs to factory defaults and reconfiguring from the start and also setting the NIDs to use Metropolia’s NTP server for accurate time.

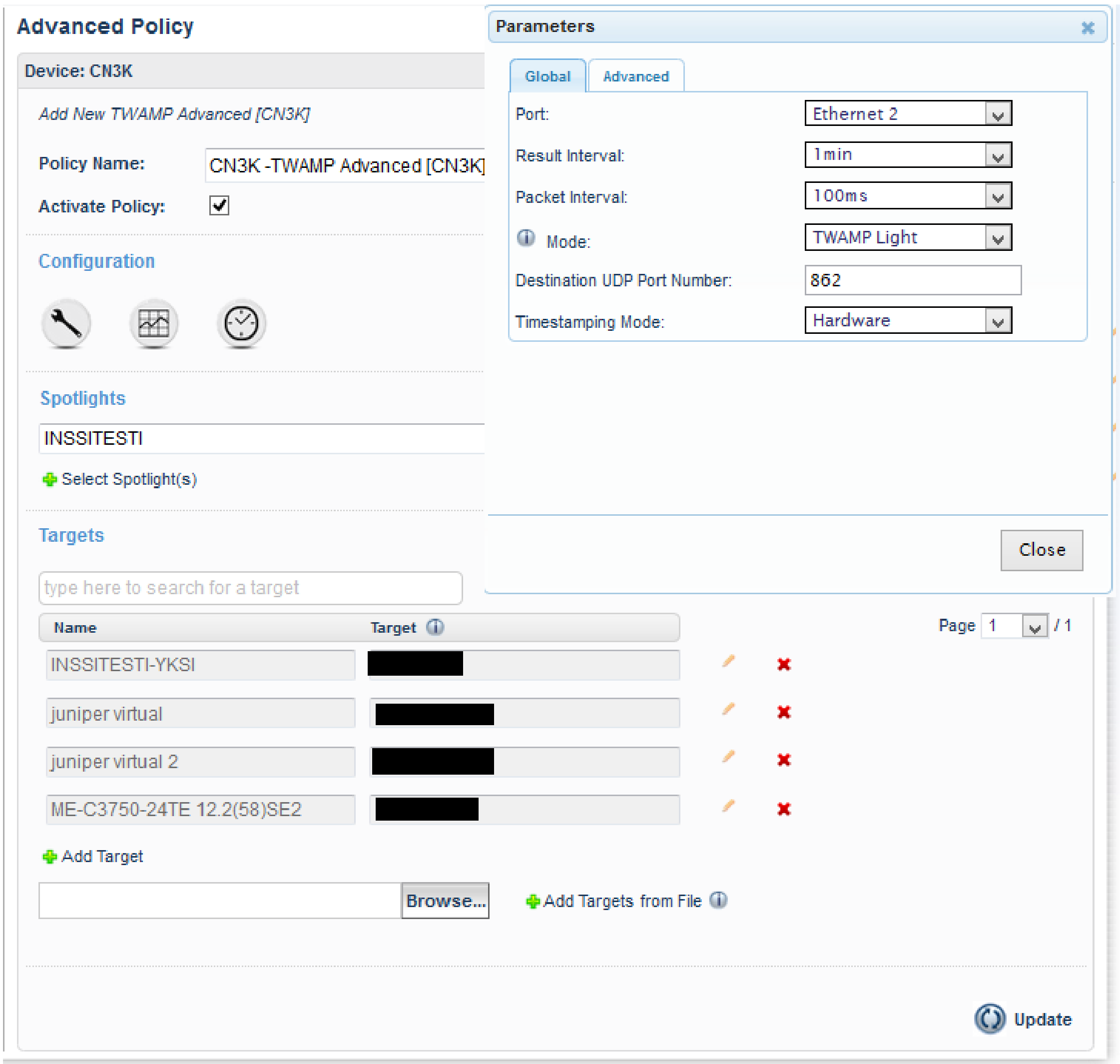
The biggest question was why some of the NIDs were working from time to time. This randomness gave the impression that the Echovault was working properly and that there was some unknown variable on the network or in the NIDs was causing them not to function properly. After trying out numerous different settings and network setups and then finally consulting Creanord it was found out that there was a bug in Echovault which was causing the problem. Fortunately there was already an update available which would fix the bug.

After getting the Echovault updated, setting up the NIDs and starting a test measurement between the devices was easy. On the NIDs there is not much to configure. One only needs to configure the interfaces with IP addresses, configure NTP and define

Echovault's IP address and password. After doing this the NID will show up in Echovault and test measurements can be configured from the policies page.

6.3 CN3K and TWAMP

Configuring TWAMP tests that are run with the CN3K are completely configured in Echovault. Like all the other measurements that are done with CN3K, the TWAMP is also configured in the advanced policy page. To add a TWAMP measurement one has to press the add button next to the host that one wants to run the tests, and then click TWAMP Advanced from the opening window. Figure 19 displays basic settings available for TWAMP.



Advanced Policy

Device: CN3K

Add New TWAMP Advanced [CN3K]

Policy Name: CN3K -TWAMP Advanced [CN3K]

Activate Policy: ☑

Configuration

Spotlights

INSSITESTI

Targets

Name	Target
INSSITESTI-YKSI	[redacted]
juniper virtual	[redacted]
juniper virtual 2	[redacted]
ME-C3750-24TE 12.2(58)SE2	[redacted]

Parameters

Port:	Ethernet 2
Result Interval:	1min
Packet Interval:	100ms
Mode:	TWAMP Light
Destination UDP Port Number:	862
Timestamping Mode:	Hardware

Figure 19. Advanced Policy – TWAMP settings

From the advanced policy device page you can access parameters, key performance indicators (KPIs), configuration history and add targets for the TWAM measurement. To get TWAMP running one has to configure port, mode, destination port and add at least one target. In the present study it was wanted to try out both TWAMP and TWAMP Light functionality so two different TWAMP advanced policies were added.

TWAMP protocol does not specify packet interval, but leaves is it up to the manufacturer to decide. Creanord's implementation allows eight different settings from 1ms to 10s. With Echovault's default settings the padding of the TWAMP packet is 41 bytes, which is the smallest allowed for TWAMP protocol. [3, p 18] Unencrypted TWAMP test package sent by session reflector is 14 bytes and padding [5, p 32]. This brings bare packet size to 55 bytes. After encapsulating in UDP (8 bytes), IPv4 (20 bytes) and Ethernet (14 bytes) frames the bandwidth needed for sending one TWAMP-test package is 97 bytes. With Echovault's 100 ms test interval 10 packets are sent every second. 97 bytes, 10 times a second in two directions is 1.94 kB/s, which is still only 0.01552% of the total capacity of 100 megabit connection. In any modern network this should not be a problem.

6.4 Cisco TWAMP

After getting the NIDs working and getting more familiar with the Echovault, the next step was to get the CN3K installed and test TWAMP with Metropolia's Cisco metro switch (ME-C3750). The switch was running Cisco IOS Version 12.2(58)SE2, that supports TWAMP. Enabling TWAMP and TWAMP-light on the Cisco IOS was a simple task. All that needed to be done was entering two commands.

```
ip sla server twamp

ip sla responder twamp
```

Example code 3. Enabling TWAMP on Cisco IOS.

After entering these two commands, the switch started to respond to TWAMP and TWAMP-light tests messages, which were sent by CN3K. At this point it was noticed that there was a weird problem with the test results. Average Two-Way Delay was peaking at around 4300 seconds or 2^32-1 = 4294967295 microseconds, and then bouncing back to around 0ms. (see figure 20)

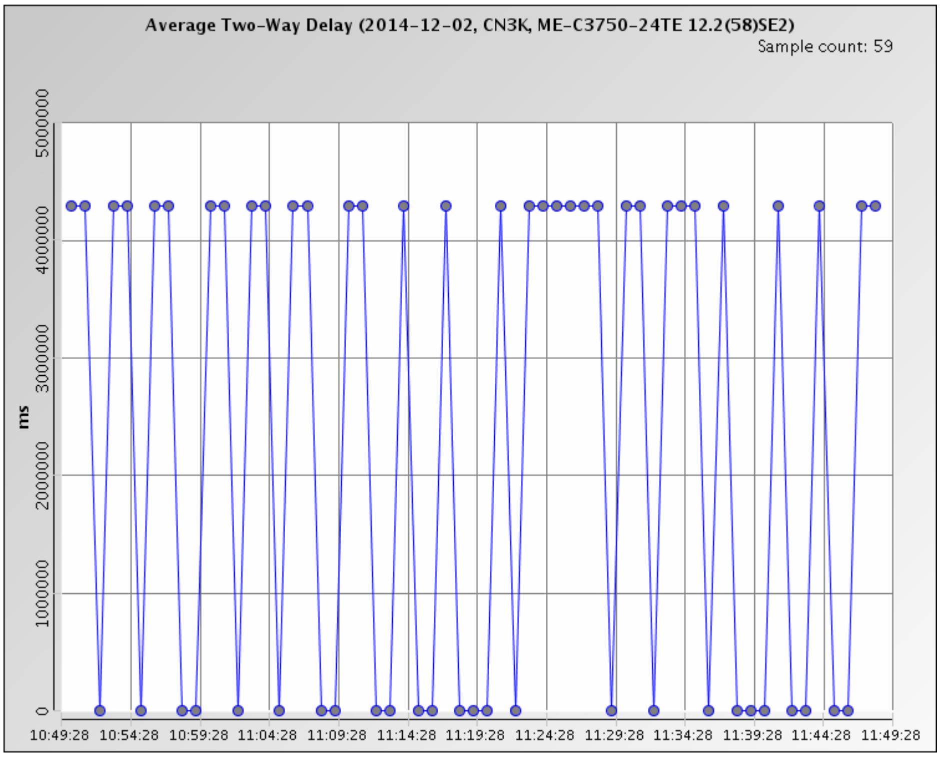


Figure 20. TWAMP - Average Two-Way Delay bug

After consulting Creanord it was found out that it was a software bug that caused the result to get the maximum value possible. The problem was in Echovault and it had already been fixed in the newer software release. After updating the Echovault and CN3K firmware to the newest version the problem changed. Instead of getting results with the maximum value of the object, the results were completely empty. The TWAMP messages were getting sent and replied. This could be seen from Packet Loss Ratio which was 0 percentage, but because of a bug somewhere the results were completely empty. Measurement was running but no data about two-way delay or any other data apart from the packet loss were getting saved or calculated. This is shown in Figure 21.

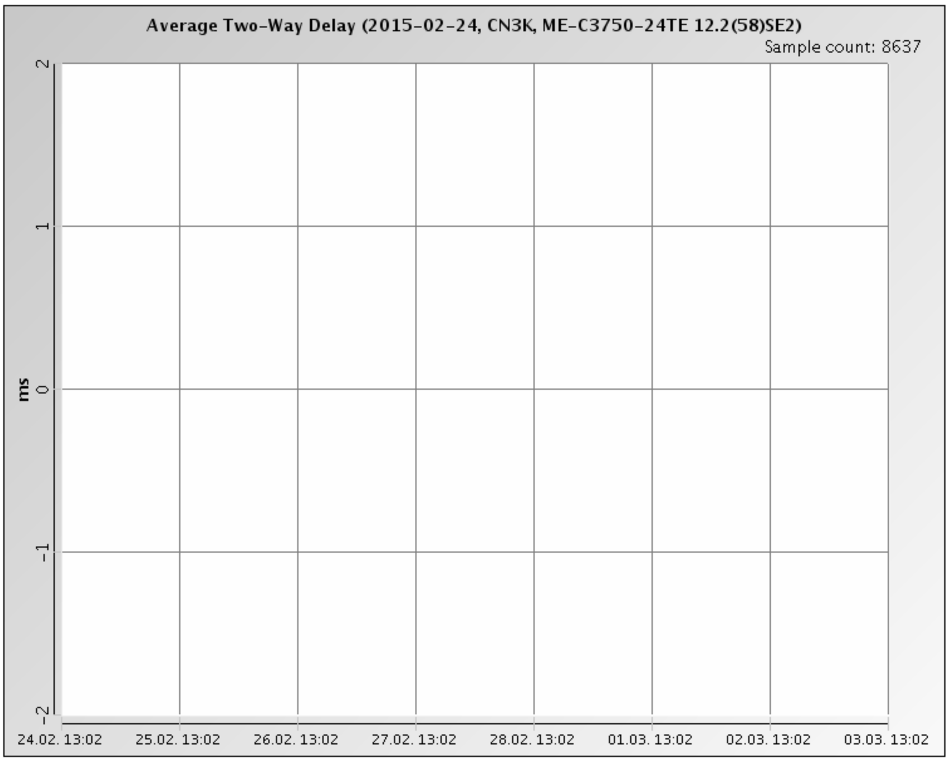


Figure 21. TWAMP – Average Two-Way Delay Bug 2.

During the time frame of the present study this problem could not be figured out. Afterwards a mail from Creanord explained that the problem could have been caused by having an incorrect port selected in the TWAMP parameters. During testing the test port was briefly changed, but it did not make difference. To figure out whether the problem was in Cisco IOS, Echovault, settings or in the CN3K firmware further research and testing would be needed.

6.1 Virtual Juniper Router and TWAMP

Juniper has a wide range of network equipment and testing TWAMP on multiple vendors was one objective of the study. However the Juniper equipment available at Metropolia did not have support for TWAMP. After being in contact with Juniper, they suggested trying a virtual version of Junos OS that was in development. Juniper provided a virtual image of Junos OS that could be run in VMware. The plan was to experiment with the virtual Junos OS and see whether TWAMP would function properly, between the virtual Junos and CN3K. At the time of testing, the virtual image of Junos was in development and not publicly available and TWAMP was something Juniper had not tested yet with the virtual image. Proper documentation was not available either at this time. These points should be taken into consideration when looking at the results.

There were a few problems with the image, which was not that surprising since the image was still in the pre-beta state. The first problem was with booting the image. With the default settings the boot process would get stuck at “Loading /boot/loader” and would not proceed no matter how long you waited. A solution for this problem was to try a different virtual disk type. After changing the virtual disk type to IDE, the system booted properly. The next problem was getting the interfaces in Junos to work and linked to the host machines interfaces. This could we solved by changing the mac address of an interface inside Junos to be the same as the mac address of the physical interface on the host machine.

Since the Junos OS was now booting correctly with working networking, the next step was to configure TWAMP and see if it would function. After configuring the TWAMP settings according to instructions from Junipers TWAMP presentation and doing other research, it was concluded that at the moment there was a problem that caused the TWAMP not to work in the test environment. Even after setting up the TWAMP-test on the virtual router and configuring CN3K through Echovault, the test results would stay completely empty and Junos OS wouldn’t show any established TWAMP connections or sessions.

The reason for this was not found out during the testing. A bug in the virtual image is quite possible, misconfiguration or incompatibility/bug between CN3K and Junos are also possible causes. A good way to start would be to test TWAMP with Juniper hardware and see if it functions with CN3K and Echovault, and if it does function properly then next try the virtual Junos OS with similar configuration and setup.

7 Discussion and Conclusions

The thesis was a long learning progress of how to use Echovault and getting familiar with all the relevant topics such as SLA, TWAMP and network testing in general. The number of businesses that need reliable internet connection is only going to increase in future, and the use of measurement protocols will get even more common. Some companies are interested in using ping for their network measurements, because in most cases it requires no additional upgrades or installations to the target host, and it is also easy and cheap to implement. Ping however has its shortcomings in accuracy and reliability, and because of that it is not the ideal solution when accurate and reliable results are desired, but in some cases it can be accurate enough to get the wanted results, but when ping is not as accurate or reliable enough there are other protocols to fill in. This is a situation where protocols such as TWAMP come in.

TWAMP uses UDP connection to perform the measurements, so it does not have the problems that ping does. It is also an open standard proposed by the IETF so manufactures are free to make their own adaption for their platforms. TWAMP support on different platforms, to which access in the present study was limited, and only available in recent versions of firmware. In the end Cisco ME-C3750 switch was the only device that worked at least at some level with CN3K and TWAMP. There is still the problem of the measurement results being empty. At this point in time TWAMP support seems to be questionable at least with the hardware that was available at Metropolia, this is mostly because Metropolia did not have switches, routers or other hardware that properly supported TWAMP. Even on the Cisco ME-C3750 there had to be a firmware upgrade performed to get it working at all. Echovault and CN3K do have support for TWAMP but without hardware that properly supports TWAMP, CN3K really cannot show its full potential as an “Ultra-Performance Network Measurement Probe”. Figuring out what is causing the problem, and additional testing with never and supported hardware could be starting point for future research.

While working with Echovault there were many problems to solve. Some of the problems were caused by lack of understanding of the Echovault and how it functions. Creanord’s employees were eager to help with any problems. On a few occasions it was even concluded that the cause of the problem was a software bug and an on-site software update was necessary. Echovault is a powerful platform for network measurements and as such it is filled with options and different menus to provide all that functionality and configurability. Echovault has quite a steep learning curve, but after getting past the initial hurdle it becomes a powerful tool at monitoring hosts and networks. During the study there were more than a few times when there were some changes made to Metropolia’s network and as a result connecting to the CN3K was not possible. At times like this it was useful to have all the measurement data available to help figuring out what had happened and when.

References

1 Jéferson C. Nobre, Lisandro Z. Granville, Alexander Clemm, Alberto Gonzalez Prieto. 2012. Decentralized detection of SLA violations using P2P technology. Online document. <http://dl.acm.org/citation.cfm?id=2499406.2499418&coll=DL&dl=ACM&CFID=357857555&CFTOKEN=15526909> (read 31.7.2014)

2 InetDaemon. 2013. What Ping is not. Online document. <http://www.inetdaemon.com/tutorials/troubleshooting/tools/ping/ping_is_not.shtm> (read 15.7.2014)

3 The Internet Engineering Task Force. 2008. A Two-Way Active Measurement Protocol (TWAMP). Online document. <http://www.rfc-base.org/rfc-5357.html> (read 22.8.2014)

4 The Internet Engineering Task Force. 1983. Echo Protocol. Online document. <http://tools.ietf.org/html/rfc862> (read 6.2.2015)

5 The Internet Engineering Task Force 2006. A One-way Active Measurement Protocol (OWAMP). Online document. <https://tools.ietf.org/html/rfc4656> (read 15.9.2015)

6 Philip Bianco, Grace A. Lewis, Paulo Merson. 2008. Service Level Agreements in Service-Oriented Architecture Environments. Online document. <http://resources.sei.cmu.edu/asset_files/technicalnote/2008_004_001_14951.pdf> (read 19.2.2015)

7 Wikipedia. 2015. Operations, administration and management. Online document. <http://en.wikipedia.org/wiki/Operations,_administration_and_management> (read 17.5.2014)

8 Brocade. 2010. OAM Best Practices in Mission-Critical MPLS, IP, and Carrier Ethernet Networks. Online document. <http://www.brocade.com/downloads/documents/best_practice_guides/oam-best-practices.pdf> (read 12.8.2014)

9 Marko Uusitalo. 2014. Ethernet OAM ja SLA monitorointi. Presentation slides. (read 18.7.2014)

10 IEEE. 2009. Station and Media Access Control Connectivity Discovery. Online document. <http://standards.ieee.org/getieee802/download/802.1AB-2009.pdf> (read 19.2.2015)

11 Cisco. 2009. Carrier Ethernet. Operations, Administration & Maintenance. Online document.

<http://conference.apnic.net/__data/assets/pdf_file/0005/58955/ethernet-oam-tutorial-final-v2_1362014627.pdf> (read 11.2.2015)

12 Cisco. 2012. Configuring Ethernet OAM (IEEE 802.3ah), CFM (IEEE 802.1ag), and E-LMI on the ML-MR-10 Card. Online document. <http://www.cisco.com/c/en/us/td/docs/optical/15000r9_0/ethernet/454/guide/45490ethernetguide/45490a_eoamonmlmr.pdf> (read 4.3.2015)

13 Brocade. 2013. Multi-Service Ironware Administration Guide. Online document. <http://www.brocade.com/downloads/documents/html_product_manuals/NI_05600_ADMIN/wwhelp/wwhimpl/common/html/wwhelp.htm#href=OAM.08.02.html&single=true> (read 21.7.2014)

14 IEEE. 2007. IEEE Standard for Local and Metropolitan Area Networks Virtual Bridged Local Area Networks Amendment 5: Connectivity Fault Management. Online document. <https://standards.ieee.org/findstds/standard/802.1ag-2007.html> (read 31.7.2014)

15 Linux Foundation. 2009. Netem. Online document. <http://www.linuxfoundation.org/collaborate/workgroups/networking/netem> (read 28.8.2014)

16 Bert Hubert. 2001. TC man page. Online document. <http://man7.org/linux/man-pages/man8/tc.8.html>(read 28.8.2014)

17 Bert Hubert. 2002. Linux Advanced Routing & Traffic Control HOWTO. Online document. <http://lartc.org/lartc.html shaping> (read 28.7.2014)

18 Alexey N. Kuznetsov. 2001. TBF man page. Online document. <http://linux.die.net/man/8/tc-tbf> (read 24.7.2014)

19 Creanord. 2015. Company description. Online document. <http://www.linkedin.com/company/creanord> (read 14.7.2014)

20 Creanord. 2013. Echovault presentation slides. (read 23.8.2014)

21 Creanord. 2015 Creanode 3000 product page. Online document. <http://www.creanord.com/products/creanode-3000-centralized-performance-probe.html> (read 11.2.2015)